Trust and Reputation Management in Decentralized Systems

A Thesis Submitted to the College of

Graduate Studies and Research

In Partial Fulfillment of the Requirements

For the Degree of Doctor of Philosophy

In the Department of Computer Science

University of Saskatchewan

Saskatoon

By

Yao Wang

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science

University of Saskatchewan

176 Thorvaldson Building

110 Science Place

Saskatoon, Saskatchewan, Canada

S7N 5C9

Abstract

In large, open and distributed systems, agents are often used to represent users and act on their behalves. Agents can provide good or bad services or act honestly or dishonestly. Trust and reputation mechanisms are used to distinguish good services from bad ones or honest agents from dishonest ones. My research is focused on trust and reputation management in decentralized systems. Compared with centralized systems, decentralized systems are more difficult and inefficient for agents to find and collect information to build trust and reputation.

In this thesis, I propose a Bayesian network-based trust model. It provides a flexible way to present differentiated trust and combine different aspects of trust that can meet agents' different needs. As a complementary element, I propose a super-agent based approach that facilitates reputation management in decentralized networks. The idea of allowing super-agents to form interest-based communities further enables flexible reputation management among groups of agents. A reward mechanism creates incentives for super-agents to contribute their resources and to be honest. As a single package, my work is able to promote effective, efficient and flexible trust and reputation management in decentralized systems.

Acknowledgements

First of all, I would like to thank my supervisor, Dr. Julita Vassileva, from the bottom of my heart for everything she has done for me. Without her, I would not be able to get through this long and thorny journey. She has been a great mentor. Not only has she spent tremendous effort and time providing guidance for my research and the writing of this thesis, but also she is a great person for being kind and encouraging, so that I could always feel strong support when I most needed it.

I would like to thank Dr. Gord McCalla, Dr. Winfried Grassmann, Dr. Michael Horsch, and Dr. Chris Zhang, for spending their precious time serving on my thesis committee and providing valuable advice.

I also would like to thank Dr. Jie Zhang for providing great collaboration and suggestions on my research.

Finally, I would like to express a very special thank to my husband, Weidong Han, and my little daughter, Cynthia Han, for their love and support.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The research on trust and reputation has attracted a lot of interest from researchers in different areas, such as e-commerce, peer to peer (P2P) networks, grid computing, semantic web, web services, and mobile networks. These systems are all open, distributed and composed of autonomous entities that may be individual persons, enterprises, services or agents that act on behalf of their users. The entities can interact with each other for various purposes, such as selling or buying products/services, sharing resources, or exchanging information. All these interactions are done through networks, which creates opportunities as well as risks. The scope of the partners that an entity can interact with is largely expanded to include nearly any one that can be reached on the Internet. These partners can use different strategies (cooperation, defection, tit for tat) to achieve their own objectives, for example, maximizing their benefits. In networks, entities present themselves by their nicknames. Although an entity's nickname can be associated with its real identity (for example, on eBay, each user's name is bound with the user's credit card number), it is often just a name randomly chosen by the entity. There is no strong connection between entities' nicknames and their real identities. An entity can also leave or join a system at any time. These freedoms can motivate some entities to cheat or defect for more profit, since they understand that they can get away with their bad behaviors. However, as a compensation for these weaknesses, information exchange through IT networks is much easier, faster and cheaper, compared with traditional methods (i.e. mail, telephone, and fax). Entities can easily gather and spread information to develop trust and reputation, which accounts for the surge of the studies on trust and reputation. Various trust and reputation mechanisms have been designed to distinguish good entities from bad ones, in order to create a safe and reliable interaction environment. The bad entities will finally be punished by losing partners to interact with.

1.1 Background

1.1.1 Trust

Trust has been studied in many disciplines, such as psychology, sociology, and economics. The views from different disciplines provide a lot of valuable observations and theories for researchers in computer science [36][43].

Psychologists tend to focus on trust as a mental attitude and study what happens in a person's mind when he trusts someone. From this perspective, Castelfranchi and Falcone [1][12] built a cognitive trust model. They view trust as a function of the strengths of different beliefs.

Sociologists study trust as a social relationship between people. This perspective is represented in computer science by the studies on multi-agent systems using social networks, for example that of Yu and Singh [86][87]. They treat a multi-agent system as a social network, where agents represent different people. Agents interact and communicate with each other, model each other, and develop trust in each other.

Economists analyze trust from the perspective of utility (cost and benefit). Game theory [89] is the most popular tool used by computer scientists to study how entities develop their trust relationship when they use different strategies, such as tit for tat, cooperation or defection. The Prisoner's Dilemma is an often used scenario.

Although researchers in computer science benefit greatly from the studies on trust in behavioural sciences, they also have to face the complexity and confusion around the notion of trust coming with the different disciplines. What is trust? The definitions of trust are so diverse that there is no common agreement even in a single discipline. Each definition of trust reflects the paradigm of a particular researcher's particular focus in her particular discipline.

In the literature, trust has been described as attitudes, beliefs, probabilities, expectations, honesty and so on. No matter what the definitions of trust are, they indicate some key factors of trust.

- Trust only exists in an uncertain and risky environment. The risk means that an interaction may result in a bad consequence, which is adverse or even harmful for an entity. For instance, in e-commerce, a bad consequence may be that a buyer loses money or a seller never gets paid. It relates to the particular partner that an entity will interact with and the partner's potential behaviours in the interaction process. For example, a seller may exaggerate his products' qualities and mislead customers to buy them at higher prices.
- Trust is used for decision making to help entities achieve desirable consequences. An example of a trust-related decision may be to select a good partner to interact with or to ask for advice from.
- Trust reflects some characteristics of someone or something, for example, trust in one's honesty. McKnight and Chervany [39] identified 16 characteristics and grouped them into

five categories as shown in Figure 1.1.

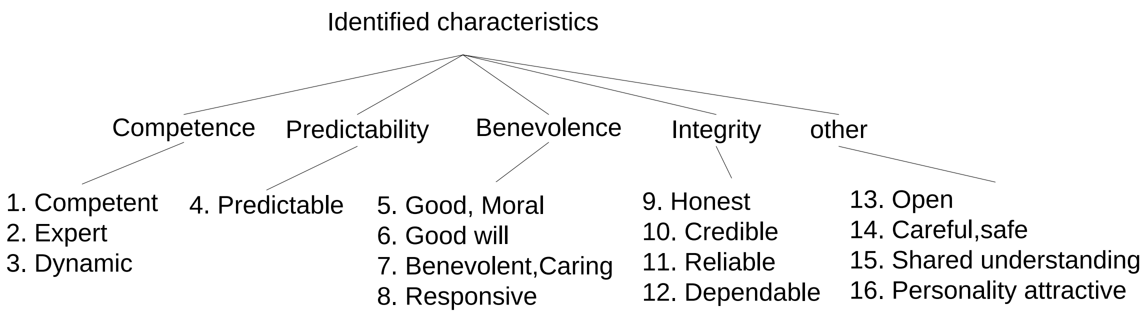


Figure 1.1 The characteristics of trust in one's honesty

- Trust is based on prior knowledge and experience, which may come from oneself or from others.
- Trust is subjective, reflecting an individual's opinion.
- Trust is dynamic and can increase or decrease with further experiences (interactions or observations). It can also decay with time [26]. New experiences are more important than old ones, since old experiences may become obsolete or irrelevant with time passing by.
- Trust is context-dependent. It is different in different contexts. For example, Mike trusts John as his doctor, but he does not trust John as a mechanic who can fix his car. So in the context of seeing a doctor, John is trustworthy. But in the context of fixing a car, John is untrustworthy.
- Trust is also multi-faceted. An entity's characteristics may be reflected by multiple aspects, each of which need to be modeled by a different kind of trust. For instance, a customer might evaluate a restaurant on several aspects, such as quality of food, price, and quality of service. For each aspect, she develops a kind of trust. The overall trust then depends on the aggregation of the trust on each aspect. While the context-specificity of trust accentuates that trust can be different in different situations, the characteristic where trust is multi-faceted emphasizes that trust needs to be modeled based on multiple aspects.

1.1.2 Reputation

Reputation is an important source for entities to acquire trust in other unfamiliar entities. It plays an essential role in large open systems, such as e-commerce, P2P computing and recommender systems, where both parties in an interaction usually do not know each other. The success of eBay, a very popular auction website, shows a good example, where the reputation values of sellers and buyers are measured by the number of positive, negative and neutral ratings that their partners in trade provide to them. The studies of Resnick and Zeckhauser [50] have shown that on eBay, 89.0% of 168,680 transactions happened between sellers and buyers who met for the first time. Almost all (98.9%) happened between sellers and buyers who have conducted transactions for no more than four times. These statistics indicate that most of the transactions happen between strangers.

Reputation is important for both parties in a transaction. For example, Bob and Peter do not know each other. Bob wants to buy a used notebook computer from Peter on eBay. Bob cannot physically examine the actual notebook before buying it. What he knows about the computer is just Peter's description of the notebook. How could Bob trust Peter about whether he has properly described the condition of the notebook? Peter's reputation comprising ratings and comments from previous partners plays a significant role. Peter's good reputation could make Bob increase his trust in Peter and conduct the transaction. For Peter, it is important to maintain good reputation, since if he has a bad reputation, he could lose potential buyers and the prices of his products might be lowered because of the loss of potential buyers.

There is not much disagreement about the definitions of reputation in the literature. Most definitions refer to reputation as the general public's opinion about an entity's character or standing [26]. Reputation is objective, representing the collective evaluation from a group of users. However, trust is subjective and reflecting an individual's opinion. As shown in Figure 1.2, the reputation of an entity can be built based on the experiences of a group of users, *user 1* to *user n*. User 1 can build his trust in an entity based on both his own experiences and the entity's reputation. He decides how to weigh them. If user 1 has a lot of experiences, he can rely on his own experiences to decide the trustworthiness of the entity or weigh his personal experiences more than the reputation. If he has few or no experiences with the entity, he may weigh the reputation more.

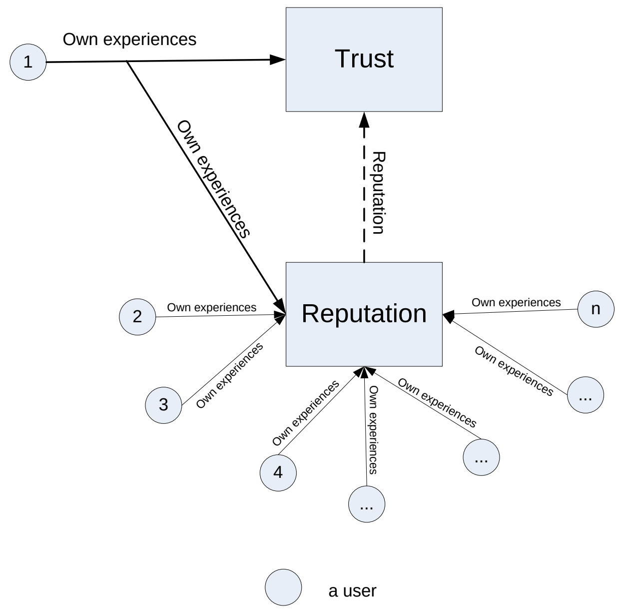


Figure 1.2 The relationship of trust and reputation

In a large open system, the benefit of reputation can be generalized from two perspectives:

- From an individual view, reputation helps entities identify good resources or trustworthy partners to interact with;
- From a collective system view, reputation encourages good behaviors of entities. Good entities are rewarded by better chances to obtain interactions. Bad entities are ultimately punished by losing chances to interact with other entities which become aware of their bad reputation.

1.1.3 Centralized vs. Decentralized

Although there are various trust and reputation systems, generally speaking, they can be classified as centralized or decentralized. Accordingly, trust and reputation mechanisms used in the two kinds of systems are different. Let us first look at an example of each kind of trust and reputation system.

A typical centralized system: eBay. eBay [74] is the largest and most popular auction website. On eBay, after each transaction, the seller and the buyer involved in the transaction can give each other a positive, negative or neutral rating, which adds 1, -1, or 0 points to their reputation, respectively. They can also leave comments about each other. Especially when people give negative ratings, they may also explain the reasons in their comments. A user's reputation is calculated as the sum of the ratings given by his/her past transaction partners in the last six months, no matter whether s/he is rated as a seller or a buyer. A newcomer starts with a reputation value 0.

Such centralized trust and reputation systems are mainly seen in the area of e-commerce. The mechanisms are relatively simple and have some common characteristics:

- A centralized entity acts as the system manager that is responsible for collecting ratings from both parties involved in an interaction.
- Users' reputation values are public and global. The reputation of a user is visible to all the other users.
- Users' reputation values are built by the system.
- Less communication is required between users. A user needs only to communicate with the centralized entity to find out the others' reputation.

A typical decentralized system: Yu & Singh's model [84]. In this kind of trust and reputation system (i.e. a peer to peer system), agents are often used to represent users and act on their behalf. In the system, there is no common place for agents to share their experiences or ratings about each other. When an agent A wants to find out the reputation of another agent B, it has to ask other agents for their ratings about agent B, including A's friends or the friends of A's friends, and then combine the ratings together; thus agent A calculates the reputation of agent B by itself.

The trust and reputation mechanisms used in decentralized systems are more complex than those applied in centralized systems. They have the following characteristics:

- There is no centralized system manager to manage trust and reputation.
- Subjective trust is explicitly developed by each agent. Each agent is responsible for developing its trust in other agents based on their direct interactions.
- No global or public reputation exists. If agent A wants to know about agent B's reputation, it has to proactively ask other agents for their ratings of B, and then synthesize the ratings together to compute agent B's reputation. The reputation of agent B developed by A is personalized because agent A can choose which agents it will ask for ratings and decide how to combine the collected ratings together to get agent B's reputation. For example, it can choose only the ratings coming from trusted agents, or it can weight differently the ratings from trusted agents, unknown agents

and even untrustworthy agents.

- A lot of communication is required by agents to exchange their ratings.

In contrast to decentralized reputation management methods, centralized methods are simpler and more efficient for agents to learn each other's reputation. However, in a centralized system, the reputation built by a central entity only reflects the general public's opinion about an agent, since all ratings about the agent are treated equally. If an agent is in a minority that does not share the rating criteria of the rest of the agents, the reputation provided by the central entity will be misleading for this agent to make decisions. In general, the aggregation of ratings in the central node leads to a loss of context information about individual preferences and criteria.

A decentralized reputation system is flexible in building agents' reputation. In the decentralized system, an agent can decide how to weigh ratings provided by different agents based on how much it trusts them. It can assign more weight to trustworthy agents and less or no weight to untrustworthy agents. The reputation built in this way is thus personalized. In a decentralized system, it is easy for an agent to develop differentiated trust in other agents based on its interests and purposes. Although collaborative filtering [8] can be used in centralized systems to provide personalized ratings for individual agents, agents cannot build differentiated trust in each other. A Bayesian network based method will be introduced in Section 1.2 to model differentiated trust in a decentralized system.

However, the decentralized method is inefficient for agents finding information to build reputation, which is represented in two aspects:

1) Agents do not know where to find reputation information. The information about an agent may be distributed anywhere in a system. Agents may not be able to find the information because they do not know which other agents have interacted with this agent for whom they are trying to build reputation.

2) Agents can be online or offline at any time. When they are offline, their ratings about other agents will not be available at the moment of request.

A super-agent based mechanism for reputation management will be introduced in Section 1.3 to solve the inefficiency problem of a decentralized trust and reputation system while keeping its flexibility.

1.2 Bayesian Network Based Trust Model

As mentioned in Section 1.1, trust is context-dependent and multi-faceted [66]. In a peer to peer file sharing system, a peer needs to develop differentiated trust in different aspects of a file provider to meet its needs in different situations. Depending on the situation, a peer may need to consider its trust in a specific aspect of a file provider's capability or in a combination of multiple aspects. I have proposed a naïve Bayesian network described in Chapter 3 to represent the probability relationships between the overall evaluation and different aspects of a file provider shown in Figure 1.3. The root node T represents a peer's evaluation on interactions with a file provider. The leaf nodes represent different aspects of the file provider. Each node is associated with some values shown as italic in Figure 1.3. One can easily obtain different probabilities of T given various conditions. The different probabilities of T represent the different trust values under different needs. For example, a peer can find out whether a file provider offers good quality of files, or whether it offers a fast download speed, or whether it offers both good quality and a fast speed.

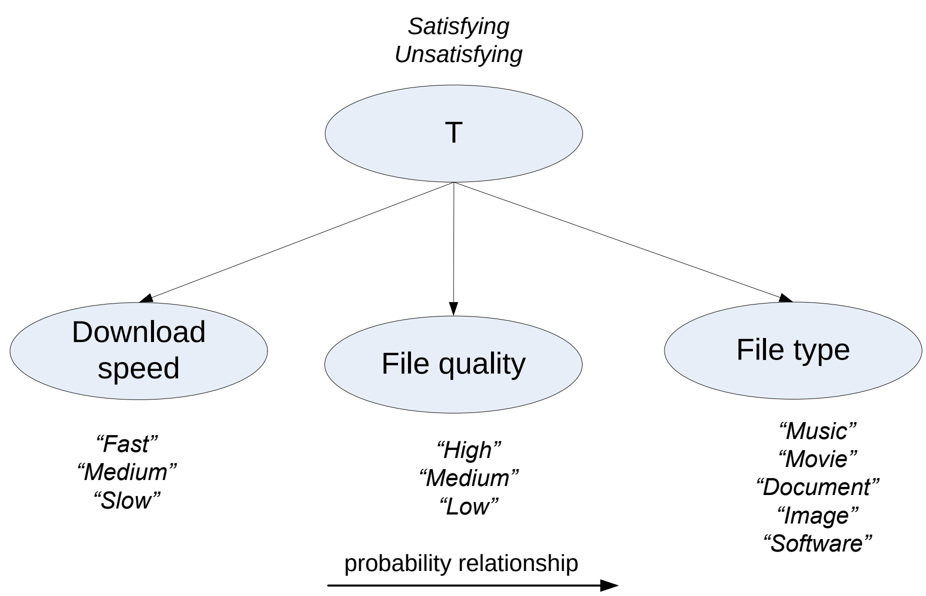
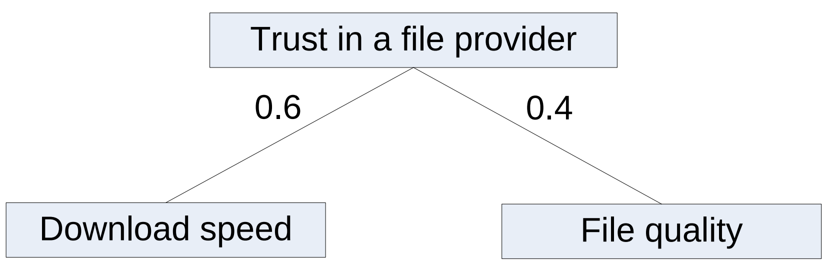


Figure 1.3 Bayesian trust

Sabater and Sierra [52][53] proposed a compositional approach to model multi-faceted trust. They adopt an ontology structure to represent trust. Figure 1.4 shows an example. Trust in each aspect is weighed by its importance. The overall trust is the sum of the weighted trust in each aspect. In their approach, the relationship between different aspects of trust is regarded as compositional. It cannot represent other non-compositional relationships. For example, a file provider can provide different types of files. It can develop trust in file types. However, it is hard to assign a weight to the trust in file types, since its relationships with the trust in download speed and the trust in file quality are not compositional. Their model is also not flexible and cannot meet peers' special needs, since it just provides a general trust in each aspect. It cannot tell you the details about each aspect. My Bayesian method makes use of Bayesian networks to represent trust in different aspects. It can not only represent complex relationships of trust, but also tell more details about each aspect based on its values, for example, whether a file provider is trustworthy at providing high/medium/low quality files or a fast/medium/slow download speed. Moreover, in Sabater and Sierra's approach, the weight of each aspect has to be provided manually. It is inconvenient to adapt to peers' different needs, and requires changing the weights constantly when peers change their needs. In my approach, peers can develop their Bayesian networks from their experiences, which provides an automatic way to adapt to peers' needs.



$$trust_{file\_provider} = 0.6 * trust_{download\_speed} + 0.4 * trust_{file\_quality}$$

Figure 1.4 Compositional trust

Chapter 3 presents the detailed Bayesian network based trust model to model differentiated trust and combine different aspects of trust. This model is demonstrated in a peer to peer file sharing application domain where file providers may provide different qualities of files and different speeds for downloading their files. It can be used to assist peers to find good providers to interact with. More specifically, the trust model allows peers to model the trustworthiness of file providers based on their previous interactions (prior knowledge) and takes into account the subjectivity of the peers. This model also allows peers to model the reputation of file providers based on recommendations about the providers shared by other peers. Experiments are carried out based a simulated file sharing system in a peer to peer network. The results confirm that modeling differentiated trust using my Bayesian network based trust model shows notable benefit.

1.3 Super-Agent Based Reputation Management

The Bayesian network based method provides a flexible way to model differentiated trust. However, the problem of inefficiency in decentralized trust and reputation systems mentioned in Section 1.1.3 still exists about finding information to build reputation. To cope with this problem, an approach for super-agent based reputation management is proposed in the thesis.

1.3.1 The Motivation

In pure P2P (peer to peer) networks, peers take equal roles and responsibilities no matter what their capabilities are (i.e. their connection bandwidth, CPU processing power or availability). However, studies [55] have shown that in practice there is a great heterogeneity in the capability of peers - between three and five orders of magnitude. Peers with poor capabilities become bottlenecks, which degrades the system performance. With the awareness of such great heterogeneity, pure P2P networks have been evolving to super-peer networks, such as Kazaa and Gnutella (v0.6). Super-peers are peers with more capabilities. Peers with poor capabilities are connected to super-peers. A super-peer acts as a server for a small group of clients (i.e. peers with poor resources) to store their information, and to send and receive messages for them [83]. Super-peers are connected with each other like peers in a pure peer to peer system to route, submit and answer queries for their clients and for themselves. Super-peer networks work more efficiently than pure P2P networks in terms of searching resources and passing messages.

Peers with sufficient capabilities may exist in other open systems, besides P2P systems, like e-commerce, grid computing, web services and multi-agent systems. With the advance of technology, easy access to the Internet and lower price for high performance computers, more and more peers can become super-peers. This research is motivated by the idea of using super-peers. For generality, peers are called “agents” because agents are often used to represent users and act on their users’ behalves, and super-peers are called super-agents (i.e. agents with sufficient capabilities). Currently, most decentralized trust and reputation mechanisms proposed in open systems have not considered the role of super-agents. Although some existing methods may be applicable to the networks where super-agents may exist, they do not take advantage of super-agents' larger capabilities. In consequence, agents with poor capabilities may become the bottlenecks for trust and reputation management just like peers with poor capabilities in P2P systems. Therefore, this research focuses on using super-agents to improve the effectiveness and efficiency of managing reputation in decentralized systems.

1.3.2 The Method

The proposed mechanism of using super-agents to manage reputation is described in Chapter 4, presented in the context of a distributed service-oriented system. Agents are assumed to provide and consume services. Super-agents are responsible for collecting information, building reputation for services and sharing the reputation information to other agents (agents with poor capabilities). Agents also send their feedback about consumed services to super-agents to help them build reputation for services. Super-agents are independent. They decide for which services they want to build reputation, which may be the services that they are interested in or the services that they want to consume in the future. Since a super-agent treats agents' feedback equally to build the reputation, the reputation of a service built by a super-agent reflects the opinion of the majority of agents who have consumed the service. It is also called the general public opinion-based reputation or the global reputation of the service. Since each super-agent is responsible to build reputation for only a subset of services, the responsibility of the central entity in a centralized system is distributed to individual super-agents.

1.4 Community Formation Mechanism

The super-agent based reputation mechanism mentioned above makes it easier for agents to find reputation information because super-agents take more responsibilities in collecting, storing information and providing the information to other agents. The reputation built by super-agents reflects the opinion of the majority of agents and can benefit agents when they are new to the system and do not have sufficient personal experience with the services to build their own trust in other agents. However, agents are different in their interests and judging criteria. The majority opinion may not fit well all agents' needs. For example, if an agent is in a minority, the reputation provided by super-agents will mislead the minority agent to make its decisions. In addition, an agent may be picky and think that the majority's opinion is not close enough to its own opinion. It wants opinions from agents with similar interests and judging criteria. In order to meet the different needs of agents, a community formation mechanism is proposed in Chapter 5 that acts as a complement to the super-agent based reputation mechanism described in Chapter 4.

In Chapter 5, a super-agent can build the general public-opinion based reputation for services according to the mechanism in Chapter 4. In addition, it can form communities based on its interests and judging criteria. It will select agents that are similar to it in terms of interests and judging criteria to join its community as its community members. In the community, the super-agent will collect evaluations about services from its community members, develop a community-based reputation for services based on the collective opinion from its community members, and share the reputation with the community members. An agent can decide whether to join a super-agent's community. If an agent thinks a community trustworthy, it will join the community. When it selects services, it will judge the trustworthiness of a service based on the service's reputation from its community. If an agent has not joined any community or does not know which community is good to join, it can use the general public opinion-based reputation to judge the trustworthiness of services.

1.5 The Reward Mechanism

Super-agents are used in both mechanisms, described in Chapter 4 and Chapter 5, responsible for providing services' reputation information. Super-agents are different from the central entity in a centralized system, which is assumed to be an authority dedicated to provide its resources (e.g. CPU power, bandwidth and storage room) and to be truthful in providing reputation information. Super-agents are not authorities. They may be selfish by nature. This brings up two problems as follows:

- Motivation: The free riding problem in P2P file sharing systems is a good example to show peers' selfishness, where peers are happy to consume resources (i.e. files) from other peers while not sharing their own resources. Super-agents are also self-interested. However, in the proposed mechanism they have to contribute more resources, collect and store reputation information, and answer queries. So super-agents need incentives for contributing resources and sharing services' reputation they built.
- Honesty: Although super-agents are supposed to be honest in providing reputation

information, they may be dishonest for various reasons. For example, a super-agent may collude with a service provider to provide false good reputation for the provider's service or bad-mouth the services from other service providers, because the super-agent can get benefits from the colluding service provider. As a result, the colluding service provider can benefit from having more customers. Therefore, how to detect dishonesty or encourage honesty of super-agents is an issue to consider.

In order to solve these two problems, a reward mechanism is designed to encourage super-agents to contribute their resources and to be honest.

Service providers will reward super-agents for building reputation for their services. If the reputation they build is used by a service consumer in evaluating a service's reputation, they will get rewards from the service's provider when the consumer uses the service. This will motivate super-agents to contribute their resources to build and share reputation information. The reward mechanism will also encourage super-agents to provide honest reputation information. If they are dishonest, service consumers will not trust them and will not use their information to judge the reputation of services. They will lose their chances of gaining rewards from service providers.

1.6 A Preview of the Thesis Contributions

In a nutshell, the purpose of this research is to promote effective, efficient and flexible trust and reputation management in decentralized systems. The contributions can be summarized as follows:

- I propose a Bayesian network-based trust model. This model provides a flexible way to present differentiated trust and combine different aspects of trust to meet agents' different needs.
- I propose a mechanism of using super-agents to manage reputation. Super-agents can build general public opinion-based reputation and provide reputation information to other agents, while other agents can help super-agents build reputation by sharing their ratings with super-agents. This mechanism helps agents find reputation information efficiently and effectively.
- I propose a mechanism for super-agents to form communities. Super-agents can build their communities to bring together agents with similar interests and judging criteria. The community can provide its community members with community-based reputation of

services that is personalized for the members' needs and can facilitate them in making good selection of services.

- A reward mechanism is designed to encourage super-agents to contribute their resources and share the reputation information they build, and also to be truthful. Super-agents can get rewards for their contributions and honest behavior. They are therefore motivated to bring their resources to the system.
- The above listed mechanisms together create a flexible unified framework to enable an efficient and effective trust and reputation management in decentralized systems. The Bayesian network based trust model enables agents to develop differentiated trust in other agents. The proposal of the super-agent based approach facilitates agents to discover reputation information efficiently and effectively. The idea of allowing super-agents to form communities and to share reputation information further enables personalized reputation management among agents with similar interests and judging criteria. The reward mechanism proposed in the framework is then a complementary element, which creates incentives for participation and honesty. As more and more agents contribute their resources and act honestly, my framework is able to create a better environment for the agents in decentralized networks.

Currently, trust and reputation mechanisms in decentralized systems have not taken full advantage of the extra power of super-agents. My research provides a way to make good use of them and holds a nice promise. With the advance of technology, easy access to the internet, and lower price, there will be more high-performance computers available in the networks, which can serve as super-agents.

1.7 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 gives a literature survey about various trust and reputation systems. A typology is used to classify these systems based on their structures. Chapter 3 introduces a Bayesian network based trust model for a decentralized P2P file sharing system, which provides a flexible method to present differentiated trust and combine different aspects of trust. Chapter 4 describes a super-agent based mechanism to manage reputation. A reward mechanism is also proposed to encourage super-agents to contribute their resources and act honestly. Chapter 5 presents a mechanism for super-agents to form communities based on their interests and judging criteria. Chapter 6 summarizes the contributions and future work.

Chapter 2

Trust and Reputation Systems: Literature Survey

This chapter gives an overview of trust on its classification and acquisition in Sections 2.1 and 2.2. It also provides a typology in Section 2.3 for classifying trust and reputation systems based on their system structures. Potential research problems are pointed out in Section 2.4, some of which lead to my proposed approaches in Chapters 3, 4 and 5.

2.1 Classification of Trust

Different kinds of trust have been described and modeled in various trust systems. Broadly, they can be categorized as individual-level trust and system-level trust. Individual-level trust is the kind of trust established and developed by a single entity. System-level trust is the kind of trust inherited from and guaranteed by the system. It is implemented by the system's inherent protocols or mechanism and mainly deals with issues related to authentication, privacy and safety of data. This section will mainly discuss various individual-level trusts. According to the relationships between a trustor (the subject that trusts a target entity) and a trustee (the target entity that is trusted), individual-level trust can be classified as follows:

Trust between a user and her/his agent(s). In multi-agent systems, agents are supposed to act and make decisions on their users' behalf. But agents might not perform as their users expect. How much users trust their agents determines how they delegate their tasks to these agents. The scenario investigated by Tang and Winoto [61] is a portfolio management system, where agents with expertise in portfolio management are used to help users invest/manage their stocks, bonds, mutual funds, etc. The trust degrees of users toward their agents directly influence the amount of money that they will delegate to their agents. In order to gain users' trust, agents need to learn users' preferences, for example, their attitudes to risk, and use their expertise to find optimal portfolio investment based on their users' preferences, as well as market situations.

Trust in providers (provision trust). Intuitively, the trust in a provider means that the provider can provide good quality products or services. For example, in P2P file sharing systems, a good provider provides both good files and fast downloading speed. But in e-commerce, the trust in a provider also means the honesty of the provider. The provider can honestly present her products' features and conditions and deliver her products. Since customers in e-commerce cannot physically see or touch products sold on web sites, they cannot judge the quality of the products by themselves. They have to rely on the information from the providers. Therefore, they are vulnerable to the providers' dishonesty, which can induce them to buy useless or inferior products for more money than the actual value of the products. So the honesty of providers along with the quality of their products and services is an important criterion for their customers to trust them in e-commerce. The eBay system is a typical example in e-commerce.

Trust in consumers. There are two kinds of situations where it is important for providers to build trust in consumers. One is the situation where providers provide services that allow consumers to access their resources, such as hard disks, CPU, and applications. The consumers need to be trustworthy, so they will not damage their providers' systems by executing harmful code, introducing viruses, accessing resources not allowed, and so on. The other situation happens in e-commerce. For example, in eBay, the consumer (the buyer) who wins an auction refuses to pay for the product he bids for, resulting in the provider (the seller) not being able to sell the product or having to spend extra time on finding another buyer. So finding trustworthy consumers is important. Providers need to be able to maintain trust in their consumers.

Trust in references (trust in information source). References are the entities that make recommendations. The trust in references measures whether an entity can provide reliable recommendations. When an entity A wants to interact with another entity B, if A is not sure about the trustworthiness of the entity B, it can ask other entities for recommendations to help it decide whether to interact with the entity B. Since entities vary in their interests and ways of judging issues, the recommendations may not all be good for an inquiring entity. There is also possibility that some references may lie for some reasons. An entity has to develop trust in these references so that it knows whose recommendations are more reliable.

Trust in groups. Groups are often formed in a society composed of heterogeneous entities according to their common interests or purposes. For example, in a P2P file sharing system, peers would like to form groups with other like-minded peers, so that they can easily discover files shared by each other that may be interesting to them. In e-commerce, it is also beneficial for buyers and sellers who are complementary to each other to form groups. By forming the groups, sellers and buyers can easily find each other, build strong relationships, and do more successful interactions. The results from Breban and Vassileva's experiments [1][64] show that both the sellers and buyers benefit from forming such groups. An entity needs to develop trust in these groups to find out the beneficial one to join. On the other side, trust in groups is also used to help an entity judge another entity according to its trust in the group that the other entity belongs to [42]. For example, a member from a reputable group, e.g. police, is often regarded as trustworthy.

2.2 Trust Acquisition

In human societies, people can gain trust through their personal experiences, such as direct interactions, observations and presumptions. They can also acquire trust from other people's experiences or from some dedicated institutions. Similarly, these methods are applied to computer systems where trust plays a role.

Direct interactions. Having direct interactions is the most reliable source for acquiring trust. It is used in most trust models. Although an entity can gain trust from different sources in different ways, the trust acquired from its direct interactions is usually weighed the most, compared with the following methods.

Observations. This method is often used in human interactions. Like in e-commerce, good descriptions and pictures of products can enhance buyers' confidence in sellers and their products. Vague and less informative descriptions or pictures will make the sellers look like suspicious, seemingly hiding something undesired [1]. Outside human interactions, this method is not so common unless a system is particularly designed for it, like robot soccer games [33] where robots can sense other robots' activities.

Presumptions. Trust can come from presumptions, which are the general beliefs derived from one's previous experiences or culture [1]. For example, products with a famous brand-name are usually regarded as being of high quality, compared with the products with no-name or unknown names. These presumptions could take various forms in human societies. It is not surprising to see that presumptions influence people's decisions in human interactions, e.g. in e-commerce. However, in multi-agent systems, it is rare for agents to have presumptions. A possible application is related to group reputation. A member from a reputable group will be presumed as trustworthy. In this way, Hales [22] shows that group reputation can also be a powerful mechanism for promotion of beneficent norms under the right conditions.

Word of mouth (reputation). Learning an entity's trustworthiness from other entities is a very effective and popular method used in most trust models and has received extensive attention from researchers. The collective evaluation about an entity, formed by aggregating the general public's opinions, is called the entity's reputation. The corresponding systems are called reputation systems [74][75]. Various reputation systems involving different algorithms and issues will be discussed in the next section.

Institutions. The trust provided by institutions can be classified as reputation-related trust and protection trust. The reputation-related trust is the trust produced from the information collected by institutions about entities' reputations, for example, the best business list published by BBB (Best Business Bureau), the best seller award given by eBay. The trustworthiness of the institutions directly influences the reliability of their information. Protection trust comes from the protections provided by the institutions against the harm committed by malicious entities. For example, eBay provides free insurance on a transaction up to a specified monetary limit. Credit card companies can also provide protections for their credit card users.

These sources of trust are not exclusive, but complementary. They may all contribute in establishing the trust of one entity in another. How to weigh these different sources of trust is a personal issue. The general practice is that before an entity has a direct interaction with a partner, it counts on the trust acquired from other entities, or institutions, or even its presumptions. After an entity gains more and more direct experiences with a partner, it can rely more on the trust it develop from direct interactions and weigh down the trust obtained in other ways.

2.3 Typology of Trust and Reputation Approaches

Various trust and reputation systems have been proposed or implemented in different open systems, e.g. P2P, multi-agent or e-commerce systems. In attempt to systematically compare the various approaches, several classifications of trust and reputation mechanisms have been proposed. They use the application areas [43], the algorithms [26], or some other characteristics [54] (e. g. a mathematical approach vs. a cognitive approach, different information sources) as criteria. But none of them allows comparing trust and reputation systems from the perspective I mention below, which is nontrivial in determining the underlying trust and reputation mechanisms.

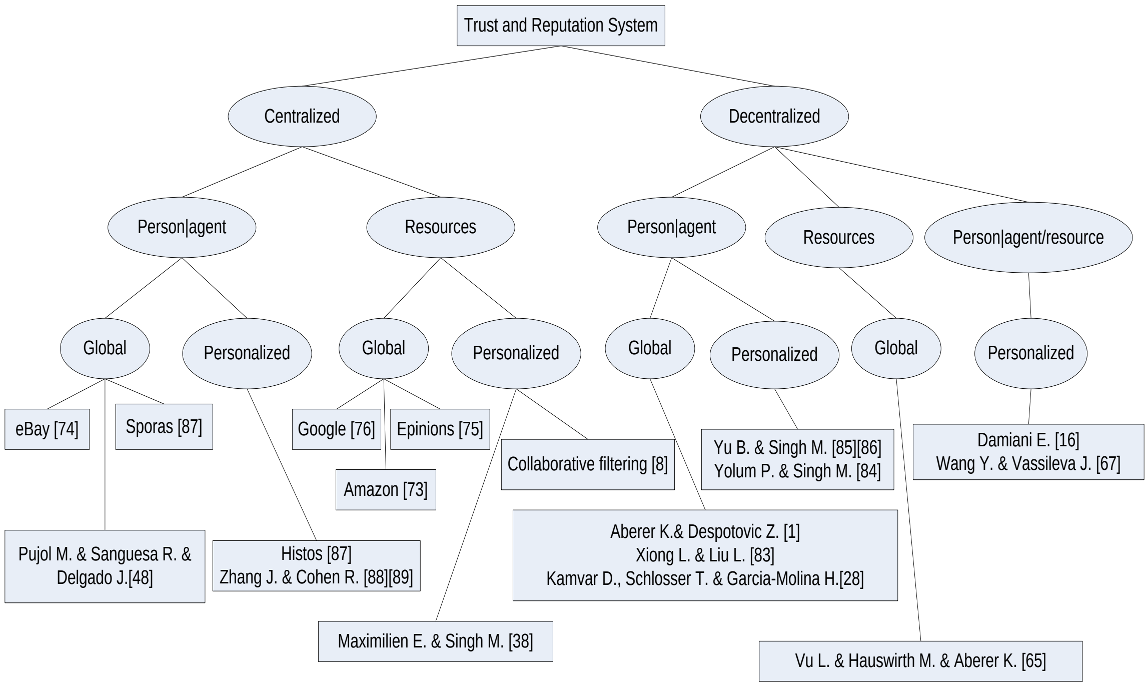


Figure 2.1 Classification of trust and reputation systems

Instead of using a flat structure, I use three criteria to analyze existing trust and reputation systems, resulting in a classification as a three-level hierarchy [68] as shown in Figure 2.1. The leaf-level represents examples of various trust and reputation systems. Each upper level in the tree is associated with one criterion used for classifying reputation systems. The three criteria are explained below.

Centralized vs. decentralized. Whether a trust and reputation system is centralized or decentralized determines the feasibility and complexity of a trust and reputation mechanism. In a centralized system, a central entity will take all the responsibilities of managing reputations for all the members. In a decentralized system, e.g. a P2P system, there is no central entity. The members in the system have to cooperate and share the responsibilities to manage reputation. Generally speaking, the mechanisms in centralized systems are less complex and easier to implement than those in decentralized systems. But they need powerful and reliable central servers and a lot of power for computing, data storage, and bandwidth for communication.

Person/agent vs. resource. Trust and reputation systems can be classified as person/agent systems or resource systems. In person/agent systems, the focus is to model the reputation of people or agents acting on behalf of people. In resource systems, the focus is to model reputation of resources, which could be products or services. Many resource systems also involve dealing with the reputation of people/agents, but it serves for the purpose of building representation of the reputation of resources. This criterion draws the line between the eBay-like reputation systems [74] and the reputation systems, such as Amazon [73], Epinions [75].

Global vs. personalized. In global reputation systems, the reputation of an entity (i.e. a person/agent/product/service) is based on the opinions from the general population, which is public and visible to all the members. In personalized reputation systems, for a particular member, the reputation of an entity is built on the opinions from a group of other members selected by the particular member. Therefore, for the particular member, the reputation of the entity is personalized. The reputation of an entity is influenced by many factors, such as the particular member's social network, environmental uncertainties, priorities, interests, preferences, etc. It is much harder and more complicated to design a global reputation mechanism in a decentralized system than in a centralized system.

2.3.1 Centralized Person/Agent Trust and Reputation Systems

Centralized person/agent trust and reputation systems are mainly seen in the area of e-commerce. These mechanisms are relatively simple. No matter whether a person's reputation is global or not, it is a function of the number of the ratings received from his/her partners. There are some common characteristics in these systems.

- A centralized node acts as the system manager responsible for collecting feedback (ratings and textual comments) from both sides in a transaction.
- A person's/agent's reputation is built by the system. Persons/agents do not need to model each other.
- Less communication is required between persons/agents. A person/agent only communicates with the centralized node to find out other persons'/agents' reputations.

2.3.1.1 eBay

eBay [74] is the largest and most popular auction Website. It has a global reputation system. After each transaction, sellers and buyers can give each other a positive, negative or neutral rating, which adds 1, -1, or 0 points to their reputations. They can also leave comments about each other. Especially when people give negative ratings, they probably will also explain the reasons in their comments. A person's reputation is calculated as the sum of the ratings given by his/her past transaction partners in the last six months, no matter whether s/he is rated as a seller or a buyer. A newcomer starts with a reputation value 0. Despite the simplicity of the reputation mechanism, empirical results [50] show that it does encourage transactions between sellers and buyers. Sellers with better reputations are more likely to sell their items. This mechanism can also prevent collusions of people artificially increasing each other's reputation. In such feedback-based reputation systems, people usually can collude in two ways. One way is that people can create fake identities and use them to give themselves high ratings. eBay allows only the two participants in a transaction to rate each other. Of course, two friends can perform dozens of fake transactions and rate each other with high ratings so that they both increase their reputation values. To prevent this, eBay charges sellers a fee for selling an item and so conducting fake transactions becomes costly. But the system does have some problems:

1) People are usually reluctant to give negative ratings since they can see each other's ratings and are afraid of revenge. In eBay, only 1% of the ratings are negative and less than 0.5% of the ratings are neutral. Neutral ratings are typically used for slightly problematic transactions, such as delays, poor communication, while negative ratings are used for serious problems in transactions. For example, the item is never shipped, arrives broken, or is different (fraudulence).
2) People can change their identities. If they get a bad reputation, they can discard their old identities, choose new ones, and start as beginners to get rid of their poor reputation.
3) In the reputation system, a person's reputation is represented by a single numeric value. It fails to convey many important subtleties of online transactions. For example, is a person's reputation built on low-value transactions or high-value transactions? Is a person reputable as a seller or as a buyer?
4) The system calculates the reputation treating all the ratings equally without taking into account the reputation of the people who give these ratings. So the ratings provided by dishonest persons are still counted.

2.3.1.2 Sporas

Sporas [88] is also a global reputation system based on feedback, similar to eBay. But it is designed against the problems of No.2 and No.4 above. To solve the problem of No.2, it uses a well-designed algorithm to calculate a person's reputation so that a person's reputation will never fall below the reputation of a beginner. Therefore, persons with bad reputations have no motivation to change their identities. In the system, two persons can only rate each other once. If they interact more than once, only the latest rating will be used. A rater's rating will be weighed by his/her reputation when it is used to calculate the reputation of his/her partner. So the ratings from persons with high reputations are weighed more than the ratings from beginners or persons with low reputations. Therefore, the problem of No.4 is solved. There is another consideration in the system. Unlike in eBay, the reputation value in Sporas cannot be increased infinitely. People with high reputation values experience smaller rating changes after each update. Compared with that in eBay, the reputation mechanism in this system is more complex.

2.3.1.3 Histos

In both eBay and Sporas [88], the reputation of a person is global, i.e. every person's inquiry about someone's reputation will obtain the same number. In human society, one's reputation is often viewed from a personal perspective. A common practice in real life of finding a person's reputation is from one's own experiences and/or asking for opinions one's friends or friends' friends, who share similar criteria or have knowledge about one's needs. In this way, a personalized bias is created on the reputation, called personalized reputation. Histos [88] is a personalized reputation system. A centralized node keeps all the recent ratings between persons and constructs a directed graph to represent ratings and persons, where nodes represent persons and weighted edges represent the most recent rating given by one person to another with arrows pointing from the rater to the rated person. Suppose there are two persons A and C. If there is an edge from A to C, which means that person A has rated person C directly, and in the eyes of person A, person C's reputation is just the rating given by himself (i.e. by A). If the two persons are not directly connected, there may be multiple paths between them. From person A's viewpoint, the reputation of person C is an average rating from the raters on these paths weighed by these raters' reputations. A recursive process is used to calculate the raters' reputations. This algorithm is more complex than that in Sporas. The time complexity is O(NM), where N is the number of the paths between the two persons and M is the average length of these paths. Each update on reputation involves a lot of computation.

All the reputation systems above are based on users' feedback toward their interaction partners. The reputation is calculated as a numerical value by combining the ratings. Such reputation systems are called feedback-based reputation systems. The advantage of this kind of reputation system is that it is very intuitive and easy to be understood and implemented. But the disadvantage is that they require the frequent involvement of users who are expected to give ratings explicitly. Ensuring a sufficient number of ratings is critical to the system. An alternative method that does not need user involvement is proposed by using the social network topology to deduce reputation.

2.3.1.4 Trust and reputation systems based on social network topology analysis

A social network is a network consisting of a group of people who are connected through various social relationships, such as acquaintance, friendship, cooperation, familial bonds, or similarity of interests. The underlying idea of deducing a member's reputation from social network topology is that reputable members in a social network tend to be well-known and highly connected nodes that can be easily identified, for example experts and people who share a lot of valuable resources. Pujol and Sanguesa [48] modeled a social network based on the information derived from users' personal web pages. This social network is represented by a directed graph, where each node represents a member and each edge represents a relationship weighed by the strength of the relationship between two connected nodes. Each node has a degree of authority. Initially, all the nodes are assumed to have the same authority. Then following the NodeRanking algorithm, the authorities of nodes are redistributed. The main idea of the NodeRanking algorithm is that the authority of each node is proportionally propagated to the nodes reached by its out-going edges, which are the edges that point to other nodes. Starting with a randomly selected node, this algorithm continues to explore other nodes in the graph using a stochastic algorithm. Finally the authorities of the nodes will converge to constant numbers, which represent these nodes' reputations. This algorithm is similar to the Pagerank algorithm [46], which is used in Google to deduce the authorities of web pages, but the NodeRanking algorithm deduces the reputation of people using the social network topology, not the topology of web page links. Although the method of using the social network topology information to calculate users' reputation does not require involvement of users, it does require the prerequisite knowledge of relationships between users on which a social network is based. How to acquire such knowledge to build a representation of the social network is critical to the success of the system. Often this knowledge is not available for a brand new system where users do not know each other. Another problem is that updating the system is tedious involving a lot of computation of updating users' reputation as well as the social network topology.

2.3.2 Centralized Resource Reputation Systems

A resource reputation system is used to build up the reputation of resources, which can serve as a guide for users to select resources. The reputation of a resource is often represented by an overall rating derived from the ratings from different users.

2.3.2.1 Epinions

Epinions [75] is a web site where users can rate and review various items, such as cars, books, movies, and computers. Users can also rate reviews in Epinions. These reviews and ratings are public. Items are organized by categories. In each category, users are classified into 5 levels, ranging from category leaders, top reviewers, advisors, most popular reviewers, to ordinary members.

- *Category leaders* are Epinions members who are in charge of a particular category. Their major responsibilities include rating new reviews, selecting top reviewers and advisors for their category and working to increase high-quality reviews for key products in their category.
- *Top* reviewers are Epinions members who write high quality reviews to help shoppers find the best products in the category and whose reviews have received the highest ratings from the Epinions community.
- *Advisors* are Epinions members who help shoppers find the best content on Epinions by rating reviews and provide constructive feedback via comments to reviewers on how to improve content quality.
- *Most popular* reviewers are Epinions members whose reviews are most popular, which is determined by the number of total visits to their reviews.
- *Ordinary members* are the default users.

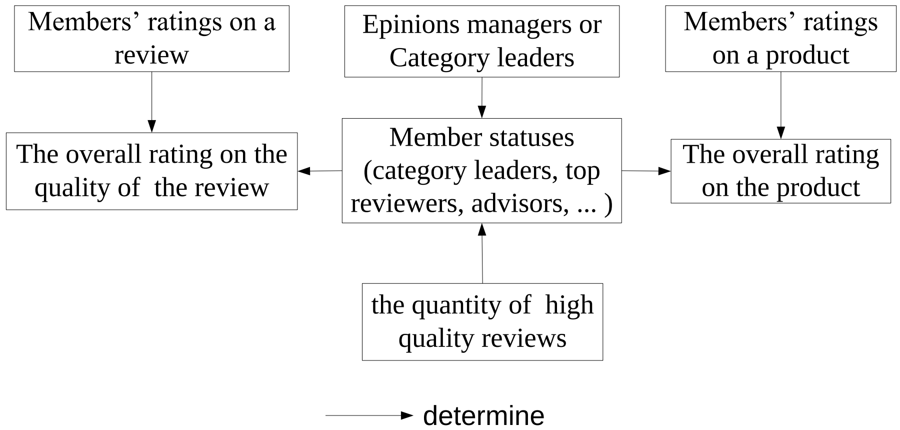


Figure 2.2 Epinions's reputation mechanism.

In Epinions, members with different statuses have different responsibilities and influences. In contrast to other websites, like Amazon [73], where all the members are treated equally, Epinions heavily depends on the member's status in the following aspects:

- The ratings on an item from members with higher status will be more heavily weighted in the overall rating for the item;
- The ratings on a review from members with higher status will be more heavily weighted in the overall rating for the review;
- The reviews from members with higher status will be placed more prominently in the review list.

Because of the importance of the member's status, a member's status is determinate by Epinions' managers or category leaders based on the quality and quantity of its reviews in a particular category. Category leaders are critical for a specific category. They can decide the other member's status, such as top reviewers or advisors. In order to be category leaders, they have to be nominated by the members in their category, and then they have to pass through interviews with Epinions managers to show they are indeed qualified. The determination of other members' statuses is conducted by the category leaders. The reputation mechanism in Epinions is shown in Figure 2.2. The overall rating on an item depends on Epinions members' ratings and their statuses.

Epinions is successful in motivating users to provide ratings and reviews. In eBay, users rate their trading partners either as a reward for good services of their partners or just for airing their opinions. In Epinions, users need stronger motivation to provide reviews, especially high quality reviews, since it requires much more effort. One possible motivation is that users want to talk about products that they love. The second possible motivation is that they want to help others and get satisfaction by sharing information about the products they consider themselves experts on. This may give them “a sense of power”, since their member status shows publicly their authority. They can also get paid for the high quality reviews they write depending on how often their reviews are read, which may be strong motivation for some users.

2.3.2.2 Collaborative filtering

Collaborative filtering [8][21] is one of the two major technologies used in recommender systems [56] to suggest items that users might like. A typical collaborative filtering system is centralized where a centralized node is responsible for collecting ratings from users and storing them in a matrix with a row for each user and a column for each item. A standard collaborative filtering algorithm has three steps [47]:

- The similarity between a given user and every other user is calculated based on the similarity of their ratings on the items that they have both rated before. The Pearson correlation is the most popular algorithm for measuring the similarity between two users.
- The predicted rating of a user on an item that she has not rated is calculated as a function of ratings from the other users who have rated the item weighed by the similarity between the user and the other users.
- The items with the highest predicted ratings are recommended to the user.

This algorithm is quite similar to that used in personalized reputation systems (see Section 2.3.3). The similarity between the two kinds of algorithm is shown in Figure 2.3. Although the algorithms of collaborative filtering systems and personalized reputation systems look similar, they are different in their focus. Collaborative filtering systems emphasize the similarity of users’ tastes, while personalized reputation systems focus on the trust between users. However, they are both used to measure the reliability of other users’ opinions/ratings. According to some definitions, trust implies similarity of users’ tastes [42][66][67] in some contexts. If two users are more similar in their tastes, they can trust each other’s opinions more. Therefore, a collaborative filtering system can be regarded as a variation of a personalized reputation system.

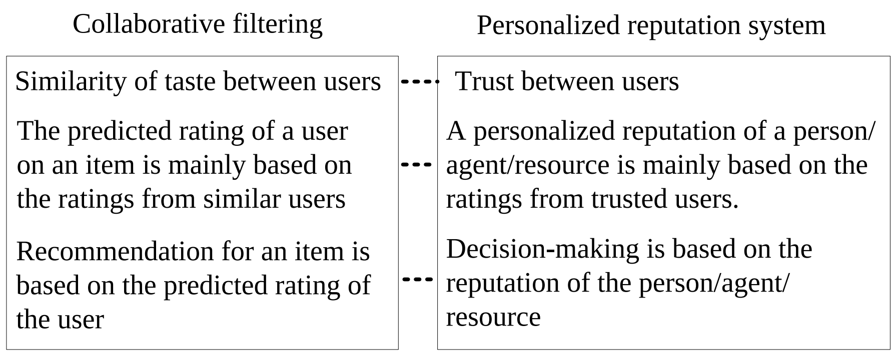


Figure 2.3 The similarities between collaborative filtering and personalized reputation systems

A collaborative filtering based recommender system can make good quality recommendations when the system has a lot of data. But it is vulnerable to the lack of data, which causes two problems, *cold start* and *data sparseness.*

The problem of c*old start* happens when new users enter a system. Since they have not rated anything, the system does not know what other users are similar to them and therefore cannot make recommendations. The cold start problem is often be solved by asking the new users to provide ratings before the system can make recommendations. In MovieLens [77], a new user will firstly be asked to rate 10 movies. However, before a new user puts effort to rate items, she needs an incentive to do so. Getting good quality recommendations is a good incentive. However, before being convinced of the usefulness of her effort, the user may not be willing to do it, which is like the chicken and egg problem.

Data sparseness is another problem for collaborative filtering. In the matrix of collaborative filtering, there may be millions of users and millions of items. A user can only rate a few items, so most of the matrix cells are empty. For example, the percentage of empty cells in the dataset of MovieLens is 95.8%. This causes a serious problem that any two users may have very low overlap in their ratings, which results in inaccuracy of prediction. Paolo and Bobby [47] suggest a method of using trust to tackle the problem. Users can explicitly express who they trust. Then trust can be propagated to other users. Say a user A trusts a user B and the user B trusts a user C. Then the user A can trusts the user C more or less. So even if the users A and C have not rated anything in common, A can still take recommendations based on C’s ratings into account. Although this method can alleviate the cold-start and the data sparseness problems, it cannot solve them. The weakness of the method is how a user finds out who he can trust among millions of other users.

2.3.3 Decentralized Person/Agent Personalized Trust and Reputation Systems

In decentralized person/agent personalized trust and reputation systems, each agent is responsible for developing its trust in other agents based on their direct interactions. No global or public reputation exists. If agent A wants to know agent B's reputation, it has to proactively ask other agents for their evaluations of B, then synthesize the evaluations together to compute agent B's reputation. The reputation of agent B developed by A is personalized because agent A can choose which agents it will ask for evaluations about B, e.g. its trustworthy friends or all known agents. Agent A can also decide how to combine the collected evaluations together to get agent B's reputation. For example, it can combine only the evaluations coming from trusted agents. Alternatively, it can weight differently the evaluations from trusted agents, unknown agents and even untrustworthy agents when combining them. In the system, agent A can get agent B's reputation based on its own knowledge of truthfulness of the agents that make recommendations for agent B. In this way, it would be hard for agent B to increase its reputation by Sybil attack where an attacker can create multiple fake identities to boost or degrade other agents' reputation. Since only agent A can see the recommendations, the references can express their feelings truthfully, not worried about potential revenges. But the tradeoff is that agents have to conduct a lot of communication and computation.

In decentralized trust and reputation systems, agents model each other and usually build two kinds of trust. One is an agent's trust in another agent's capability in providing services. The other one is the agent's trust in another agent's ability in providing recommendations. In Yu and Singh's model [87], the two kinds of trust are called respectively an agent's expertise and sociability. An agent's expertise refers to the agent's ability to provide required services. An agent's sociability is the agent's ability of giving good referrals, i.e. being able to suggest agents that can provide the required service. For example, in Figure 2.4, when agent A sends a service query to agent B, agent B recommends agent C, and then agent C recommends agent D, and finally agent D recommends agent E, the agent providing the service, agents B, C, and D are all referrals. The agent's overall trust in another agent is just the linear combination of the agent's trust in the expertise and sociability of another agent.

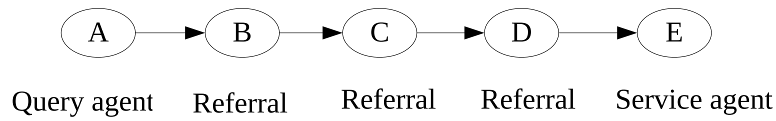


Figure 2.4 Referral graph.

In the model [66] of trust and reputation in peer-to-peer file sharing systems, the second kind of trust is used to judge the reliability of recommendations from other agents, which can help the querying agent decide whose recommendation should be considered and whose should not. The reliability of an agent's recommendation includes two aspects, truthfulness and similarity. Truthfulness means whether an agent is honest in telling its recommendations. Similarity implies whether two agents are similar in preferences and ways of judging issues. Since agents are heterogeneous, if they are not like-minded, they may disagree with each other frequently. One's recommendation is of no use for the other. If two agents are similar in this sense, their recommendations may be valuable for each other.

The procedure in these models usually follows these six steps:

1) ***Send queries.*** An agent starts a query for a specific service. When other agents receive the query, they will check whether they can provide the required service. If yes, they will send answer messages to the inquiring agent to tell it that they provide the required service. Otherwise, they just simply forward the query.
2) ***Ask for recommendations.*** When the inquiring agent receives the answer messages, it can get a list of providers that offer the required service. Then it can choose one of the providers that it trusts most. If the agent cannot decide which provider is trustworthy to provide good services, it can send another query to ask about the trustworthiness of these providers. If the agents that receive the query have interacted with these providers before, they can send recommendations to the inquiring agent.
3) ***Select provider.*** According to the received recommendations and its own experience, the inquiring agent can decide from whom to get the service. How an agent weighs the recommendations of other agents and its own experience to make a decision is an open question, related to the agent's subjective way of managing trust. Some agents may prefer to trust their own experiences more even if they had very few interactions with the service provider. Others may rely entirely on the recommendations.

4) ***Interaction.*** The inquiring agent gets the service and evaluates it.
5) ***Update trust.*** Based on the result of the interaction, the inquiring agent updates its trust in the provider's ability to provide good services and also the trust in the agents who provided recommendations.
6) ***Update neighbors.*** The inquiring agent selects agents that it trusts most as its neighbors in terms of the overall trust, which usually is a linear combination of the two kinds of trust, the trust in an agent as a service provider and the trust in the agent as a recommender. In Yu and Singh's model, this step is before the step of sending a query. After an agent generates a query, it will decide to which agents it will send the query by considering the agents' abilities in answering the query and their sociality learned from previous experiences.

In a decentralized trust and reputation system, the agents gather information and learn the reputation of others using social networks. Agents build up their social networks by learning. Initially agents just randomly connect with each other without knowing anything about other agents. After some interactions, they know which agents are trustworthy and which are not. Then they select the agents that they think are trustworthy as their "neighbors" in the social networks. An agent's social network is a network that starts from the agent and is extended to its trustworthy agents and their trustworthy agents, and so on. The entire social network of a system with many agents is represented as a network where some agents may be highly connected by trust relationships and some may be isolated. This is because agents that provide good services are gradually recognized by more agents as trustworthy and connected, while the agents that offer poor services will eventually lose connections from other agents. In [85], Yolum and Singh investigated the structure of the social network proposed by Yu and Singh [86]. They used *PageRank*, the metric used by Google to rank web pages, to measure the authoritativeness of agents. If an agent has more agents connected to it and these agents have high authoritativeness, the agent will have a high authoritativeness and a high *PageRank.* Yolum and Singh show that the percentage of agents with high *PageRanks* and the variance of the *PageRank* values are influenced not only by the percentage of agents with high expertise, but also by the referral selection policy, i.e. the way an agent weights an agent's expertise and sociability, and the policy of whom an agent asks for recommendations, for example, asking all the referrals, asking some referrals, or asking the best referral.

2.3.4 Decentralized Person/Agent Global Reputation Systems

In the systems discussed in the previous sections, in order to know the other agent's reputation, an agent has to ask around, collect evidence about the trustworthiness of the other agent, and calculate the reputation of the other agent by itself according to the evidence collected. The calculated reputation is personalized and only visible to the agent that does the calculation. If several agents want to find the reputation of the same agent, each of them has to repeat the same procedure to calculate its individual reputation representation of the agent in question. It is possible to build instead a decentralized reputation system where an agent's reputation is public and global, showing how all agents trust an agent. Such systems are proposed in [1][2][28][84], called decentralized global reputation systems. Two major questions are addressed in the systems: how to manage reputation storage and how to calculate an agent's global reputation.

2.3.4.1 How to manage reputation storage

Unlike in a centralized system where the central node acts as an authority to calculate and store agents' reputations, in a decentralized system, there is no central node. The task of a central node is carried by all agents, so each agent is responsible for calculating and storing some agents' reputations. An agent's reputation should be also calculated and stored by several agents independently to ensure redundancy in case of existence of malicious agents who can mislead other agents about reputations of agents that they are responsible for. Who is responsible to handle whose reputation? Two methods have been proposed.

- *Distributed hash tables (DHT).* Multiple hash functions are used to map a single agent ID to several positions where corresponding agents will calculate and store this agent's reputation individually [28].
- *P-Grid* [1][2][84]. The idea of this method is to divide the whole big set of agents into small subsets in a top-down way according to the structure of a binary search tree. Initially, all the agents belong to a big set. Each of them is responsible for managing all the agents' reputations. Each agent in the same set has the same responsibility. When two agents in the same subset meet, they can divide the subset that they belong to into two sub-subsets as shown in Figure 2.5 (a). At the same time, they divide their responsibility into half so that each agent only handles the reputations of half of the agents in the set. Each agent also keeps a reference to an agent in the other sub-subset

in its routing table in order to cover the search space of the other sub-subset. Agents in the same subset will individually calculate and store the reputations of the agents whose index prefixes match the search path of the subset. An agent's index is a binary string encoded from its ID. For example, in Figure 2.5 (b), agent 3's index is *011*. It is responsible for calculating and storing the reputations of agents whose indices have a prefix 10, which are agent 4 and 5 denoted by P4 and P5, whose indices are *100* and *101* respectively. According to the routing table, agent 3 is also responsible for forwarding queries for agents with an index prefix *0* to agent 2 (P2) and agents with an index prefix *11* to agent 5 (P5).

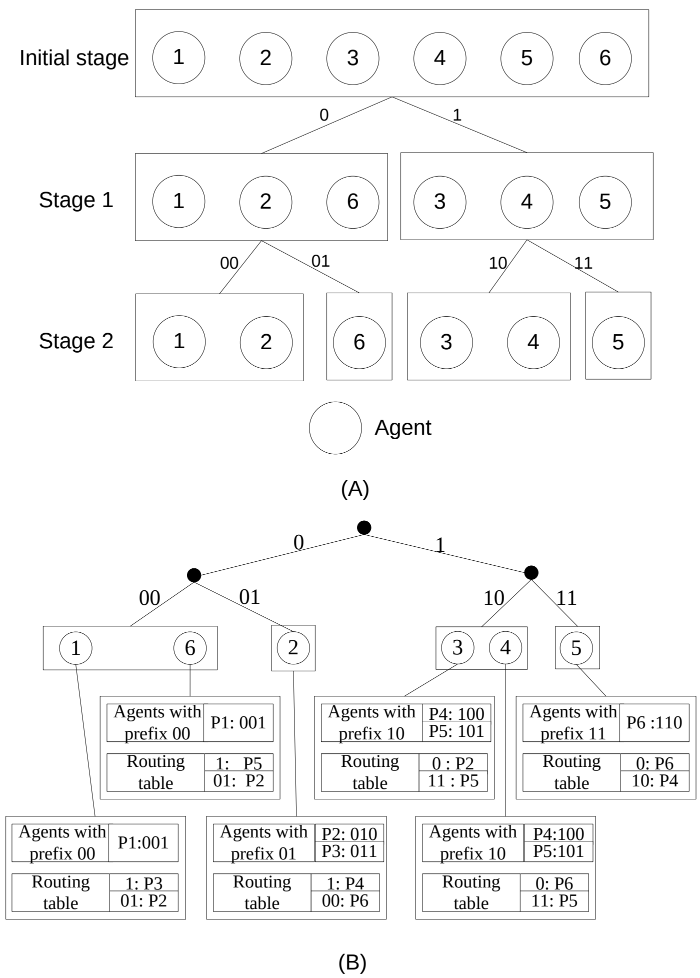


Figure 2.5 An example of P-Grid.

2.3.4.2 How to calculate an agent's global reputation

In a decentralized global reputation system, each agent's global reputation is computed and stored by some other agents designated by the method of DHT or P-Grid. Agent A is called agent B's reputation manager if agent A is responsible for computing and storing agent B's global reputation. In order to calculate an agent's global reputation, the agent's reputation manager has to aggregate all the other agents' evaluations in the target agent and weigh these agents' evaluations by their reputations. Initially, all the agents start with a default reputation. Suppose there are three agents, A, B, C. Agent B is a service provider. Agent C is agent B's reputation manager. After A gets a service from B, it can tell C its evaluation about B. Then C will re-compute B's reputation based on A's evaluation and propagate the new reputation to other agents. Other agents will also update the reputations that they are responsible for based on the received reputation of B and then propagate the new computed reputations to others. This procedure goes on until agents' reputation values converge. The whole process involves a lot of iterations. Compared with the algorithms used for calculating personalized reputations, computing agents' global reputation is much more complicated and expensive and involves a lot of communication and calculation. Although Kamvar [28] and Xiong [84] suggested some ways to reduce the high cost of this algorithm in communication and calculation, such as using cache mechanisms or predefining some trustworthy agents, this algorithm is hard to implement in real systems. There are still some problems unaddressed. For example, there are multiple agents responsible for a single agent's reputation in the system. They may reach different reputation values about the same agent based on the information they receive individually. The problem is how to decide which reputation values are more reliable. Aberer and Despotovic [2] have addressed this problem in their complaint-based reputation system, where an agent's reputation managers are responsible for collecting other agents' complaints about the agent. If there are more complaints against an agent, the agent's reputation will be worse. Since the agent's reputation managers may be offline or online, the number of complaints that they have received may be different. If agent A wants to know agent B's reputation, it may get several different answers about agent B's reputation, i.e. a different number of complaints. Then agent A will weigh these numbers by the frequencies that these reputation managers appear to be online during a period of time. The assumption is that the numbers reported by the reputation managers that have been more often offline is less reliable and therefore should be weighed less.

2.3.5 Decentralized Person/Agent Resource Personalized Reputation Systems

As in centralized systems, agents in decentralized systems care about both good agents and good resources. The reason for building reputation of resources is that when agents search for resources, they usually get a long list of resources that match their requirements. Some of them may be bad, damaged, or even dangerous, for example, files with viruses. A public representation of the reputations of the resources can prevent agents from selecting bad resources and bad resources from being propagated in the networks, since even honest agents sometimes may share bad resources without knowledge. In Damiani et al.'s [16] and Wang and Vassileva's reputation systems [67], reputations of resources are explicitly computed for P2P file sharing systems.

In Damiani et al.'s approach [16], an agent searches for resources using the standard Gnutella protocol. After the agent gets as a result a list of resources and their providers, it will send another query to ask for evaluations not only about the agents who provide the resources, but also about the resources themselves. After getting the evaluations, the agent will go further to check these evaluations' truthfulness through a series of steps. Then, after aggregating all the valid evaluations, the agent can finally make a decision from which agent to download which resources. The authors do not propose an exact algorithm for aggregating and calculating agents' or resources' reputation. They focus on various attacks that could happen in such reputation systems and the ways to prevent them.

In contrast to Damiani's approach, where agents have to discover other agents' reputation or resources' reputation by themselves, Wang and Vassileva [67] use communities to facilitate agents to discover reputation of agents or resources. In their approach, agents will self-organize to form communities according to their common interests. A community is an organization consisting of multiple creators and agents. Creators are self-selected agents who have high computing ability and bandwidth. They organize other agents with common interests together and collect information from these agents, including their ratings about each other and ratings about resources shared among them. Then the creators will aggregate all the information together and generate an overview of agents' reputations and resources' reputations. So from the community's creators, agents can quickly find out what resources are shared in the community and whether these resources or agents are good or not. In this approach, the reputations of agents and resources are calculated as the average of received ratings. Figure 2.6 shows the structure of a community. There are three lists, the creator list, the agent list, and the paper list. The creator list stores the information about the creators. The agent list includes the information about agents and their reputations. The paper list provides information about papers and their ratings.

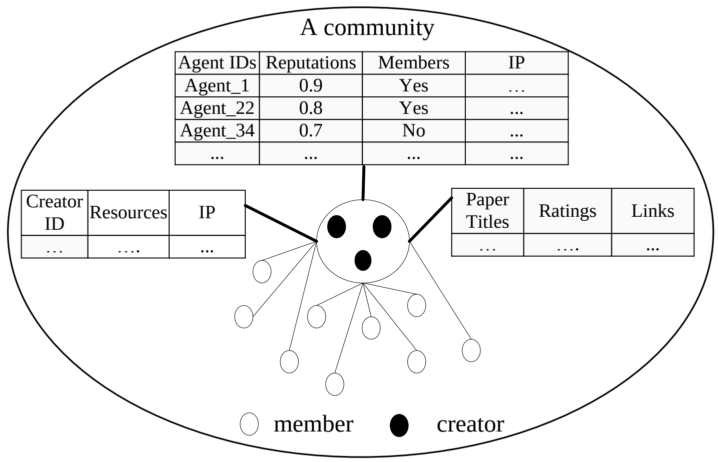


Figure 2.6 The structure of a community.

2.4 Research Problems

As more and more trust and reputation mechanisms are suggested and applied in real systems, some new problems are coming up. In this section, I will discuss these new issues.

2.4.1 Dealing with Unfair Ratings

Generally, many trust and reputation systems use ratings from users/agents (raters) [74][86][84]. The value of trust or reputation is often calculated as a function of integrating ratings from different raters. How to integrate these ratings is important. If there is an incentive in the system, one can't expect that all the ratings are sincere. Raters are heterogeneous. Some raters may be honest and provide ratings according to their true experiences. Some may be malicious. They give others unfair ratings on purpose for their own advantage. For example, in eBay, the retaliation from sellers is obvious. If a buyer gives a negative rating to a seller, the seller will also give a negative rating to the buyer even though the buyer behaves well in the transaction. So how to detect malicious raters' unfair ratings and prevent their bad influence on one's reputation is gaining more and more attention from researchers. In [58], Sen and Sajja use the majority rating from the raters to decide one's reputation under the assumption that most of raters are honest. Under this assumption, the algorithm can efficiently eliminate unfair ratings from the liars. But how to validate this assumption in a system is a problem. Another weakness of this algorithm is that it can only deal with the binary case where ratings are either high or low. In [17], Dellarocas focuses on filtering out unfairly positive ratings by combining the nearest neighbor algorithm [8] with Macnaughton-Smith's clustering algorithm [34]. However, he does not provide a solution to the problem of how to deal with unfairly negative ratings. Although the approach suggested by Whitby, Jøsang and Indulska [72] can filter out both unfairly positive and negative ratings, their assumption is that dishonest raters' behaviors are consistent, i.e. dishonest raters always provide unfair ratings. If raters sometimes rate fairly and sometimes not, the approach may not work properly.

The common characteristic of these previous methods is that they are all based on statistical analysis and comparison of ratings themselves. Another approach to dealing with unfair ratings is to use the raters' reputations [28][88]. The assumption is that raters with low reputations are more likely to give unfair ratings than those with high reputations and therefore their ratings should be weighed less.

In summary, all existing approaches depend on specific assumptions and cannot fully combat the problem of unfair ratings. Combining both the statistical method and the method of using raters' reputation, the reliability of trust and reputation systems may be improved.

2.4.2 Comparison of Different Trust and Reputation Mechanisms

With the surge of various trust and reputation mechanisms, different algorithms and performance metrics have been suggested to evaluate these systems. However, there is no unified way to compare and measure these different mechanisms and find out which one is better. Some general metrics have been pointed out in [30][37].

- Effective. The probability that a user/agent can locate trustworthy interaction partners should be high.
- Efficient. The computation and communication cost should be as small as possible.
- Adaptive. The mechanism should be able to adapt to different changes of behaviors of users/agents who may suddenly lose their competences or act maliciously.
- Robust. The mechanism should be able to detect and prevent various attacks and threats, such as pseudo spoofing attack where an attacker can create and control multiple fake identities to have a large influence on other agents' reputation, shilling

attack where an attacker can create and control several real identities to artificially increase or decrease other agents' reputation, and man in the middle attack where an attack can intercept other agents' messages and modify them [16].

- Quickly converging. When new users/agents join the system, others should be able to detect their trustworthiness as soon as possible.
- Multidimensional. The mechanism should be able to distinguish and manage users/agents' different trusts or reputations in multiple categories.

2.4.3 Interdisciplinary Problems

As mentioned in Section 1.1, the studies of trust and reputation in the area of computer science have benefited a lot from the studies in other areas, such as psychology, sociology, and economics. But when more and more trust and reputation mechanisms are applied in online systems, they begin to influence people's behaviors, e.g. their decisions on buying products, and public opinion formation. In turn, this brings many new questions to the studies in psychology, sociology, economy, and computer science as well, for example, the impact of these mechanisms on economic efficiency, social fairness, and users' strategies and risks. These questions have to be answered by integrating the knowledge in multiple areas. Dellarocas and Resnick point out some interesting interdisciplinary problems in [18] for further study. One is the interaction of online and offline reputation. Most e-commerce reputation systems are built for selling or recommending products, where the reputation of products mainly depends on users' ratings. In real markets, the reputation of products can also be indicated by their brand names, local stores, advertisements or third parties. It is still unknown how these two kinds of reputations interact and influence users' decisions. There is another issue about how to motivate users to make contribution, such as providing ratings or resources. This issue exists in most reputation systems, especially in rating-based reputation systems, which is crucial for their practical deployment. Beenen [6] uses a social psychology-based method to make users feel important and unique and therefore contribute more. Vassileva [62] has introduced and studied this problem with her students Bretzke [9] and Cheng [13], and proposed methods to motivate users by allowing them to develop personalized relationships with other users, visualizing their contributions, and giving them status and recognition in their community. Some other interdisciplinary issues include privacy, users' emotion, and consensus about key concepts related to trust and reputation, and so on.

2.5 Summary

In summary, many trust and reputation systems have been developed. In this chapter, I have presented a systematic overview of these diverse state-of-the-art approaches. I first gave an introduction of trust on its classification and acquisition in Sections 2.1 and 2.2. Trust can be classified based on the relationships between a trustor and a trustee. It can be acquired from different sources. My Bayesian network based trust model in a P2P file sharing network that will be described in Chapter 3 focuses on building trust in file providers based on direct interactions as well as recommendations provided by others (references). The trust in these references is also evaluated by my trust model to cope with the unfair rating problem mentioned in Section 2.4.1. My Bayesian network based trust modeling can also be used to tackle the multi-dimensional problem of trust mentioned in Section 2.4.2. It provides an advanced method to manage users/agents' differentiated trust.

In Section 2.3, I also provided a typology for classifying trust and reputation systems based on their system structures. I discussed the difference between centralized and decentralized trust and reputation systems, and their advantages and disadvantages. This motivates my proposal of using super-agents to manage reputation in Chapter 4 and to form communities in Chapter 5. This approach is semi-centralized, or more specifically, emergent centralization in a decentralized system through self-organization.

A reward mechanism will be described in Chapters 4 and 5 to deal with the unfair rating problem and the motivation problem mentioned in Sections 2.4.1 and 2.4.3. Service providers will provide rewards to super-agents/agents for sharing their ratings about the providers' services. If their ratings are used by a service consumer in evaluating a service's reputation, they may get rewards from the service's provider when the consumer uses the service. This will motivate super-agents to contribute their resources to collect ratings and share their ratings. The reward mechanism will also encourage super-agents to provide honest ratings. If they are dishonest, service consumers will not trust them and avoid using their ratings to judge the reputation of services. They will lose chances to get rewards from service providers.

Chapter 3
Bayesian Network-Based Trust Model

P2P networks are networks in which all peers cooperate with each other to perform a critical function in a decentralized manner. All peers are both consumers and providers of resources. Compared with a centralized system, a P2P system provides an easy way to aggregate large amounts of resources residing on the Internet or in ad-hoc networks with a low cost of system maintenance. The decentralization, however, also causes some problems. Since peers are heterogeneous, some peers might be benevolent in providing services. Some might be buggy and cannot provide the services they advertise. Some might be malicious by providing bad services. There is no centralized entity to serve as an authority to supervise peers' behaviors and punish peers that behave badly. Malicious peers may have an incentive to harm other peers if they can get a benefit, because they can get away with it. Some traditional security techniques, such as service providers requiring access authorization, or consumers requiring server authentication, are used as protection from known malicious peers. However, they cannot protect from peers providing variable-quality service, or peers that are unknown. Mechanisms for trust and reputation can be used to help peers distinguish good partners from bad ones. This chapter describes a trust and reputation mechanism that allows peers to discover partners who meet their individual requirements, through individual experience and experiences shared by other peers with similar preferences.

The rest of this chapter is organized as follows. Section 3.1 introduces my approach – a Bayesian network-based trust model and a method for building reputation based on recommendations. The experimental design and results are presented in Section 3.2. Section 3.3 discusses related work on trust and reputation. A summary is presented in the last section.

3.1 Bayesian Network-Based Trust Model

A P2P file sharing application is used as an example in the discussion. However, the model is general and can be applied to other applications, like web-services, e-commerce, recommender systems or P2P distributed computing.

3.1.1 The Overview of the Trust Model

In the area of file sharing in P2P networks, all the peers are both providers and users of shared files. Each peer plays two roles, the role of file provider offering files to other peers and the role of user searching and downloading files provided by other peers. In order to distinguish the two roles of each peer, in the rest of chapter, when a peer acts as a file provider, it is called file provider; otherwise, it is simply called peer. Peers will develop two kinds of trust: the trust in the file providers' capability (in providing files) and the trust in the other peers' reliability in making recommendations. Here the reliability includes two aspects:

- Truthfulness – whether a peer is truthful in telling its information.
- Similarity – whether a peer is similar to the peer requesting the recommendation in preferences and ways of judging issues.

A peer's reliability as a referee depends on both being truthful and similar in its preferences to the peer requesting the recommendation. Since peers are heterogeneous, they may have different preferences and judge issues based on different criteria. For example, some peers may consider a movie provider good because it provides movies with high quality, while others may consider the movie provider bad because the download speed is very slow. If two peers A and B are similar in their evaluation criteria, peer A can trust B's recommendations, if it knows that B is truthful. However, if the peers have different evaluation criteria, peer A cannot trust B's recommendations even when B tells the truth.

It is important for a peer to develop trust in other peers as references in a decentralized system. When a peer is not sure about the trustworthiness of a service provider, it can ask only a few most trusted peers for recommendations instead of asking a large number of peers. This not only helps the peer get more reliable recommendations, but also saves time and communication costs.

A search request in file sharing P2P applications usually results in a long list of providers. If a peer happens to select a file provider with bad quality or slow download speed, the peer will waste time and effort, which may lead to the peer's frustration and abandoning the system. In order to solve the problem, the mechanism of trust and reputation is used as shown in Figure 3.1. Once a peer receives a list of file providers for a given search, it can arrange the list according to its trust in these file providers. Then the peer chooses one of the file providers on top of the list. If the file provider is trustworthy according to the peer's previous experiences, the peer will interact with the file provider (download files). If the peer is not sure about the trustworthiness of the file provider, for example, the peer has no interactions or only a few interactions with the file provider, it can ask other peers to make recommendations about the file provider. How the peer uses the reputation of the file provider and its own trust to make a decision with which file provider to interact depends on the peer's goals and preferences. Some peers may prefer to trust their own experience and rely on their trust even if they had very few interactions with the service provider. Others may be more cautious and rely on the reputation of the service provider. After each interaction, the peer updates its trust in the file provider according to its evaluation of the interaction. If the interaction is satisfying, it will increase its trust in the file provider; if the interaction is not satisfying, it will decrease its trust in the file provider. If the decision for interaction is based on the reputation of the file provider, derived from other peers' recommendations, the peer will also update its trust in each of the peers that give recommendations (these peers are called "referees"). If a referee's recommendation is consistent with the peer's evaluation of the interaction, the peer will increase its trust in the referee; otherwise, it will decrease its trust in the referee.

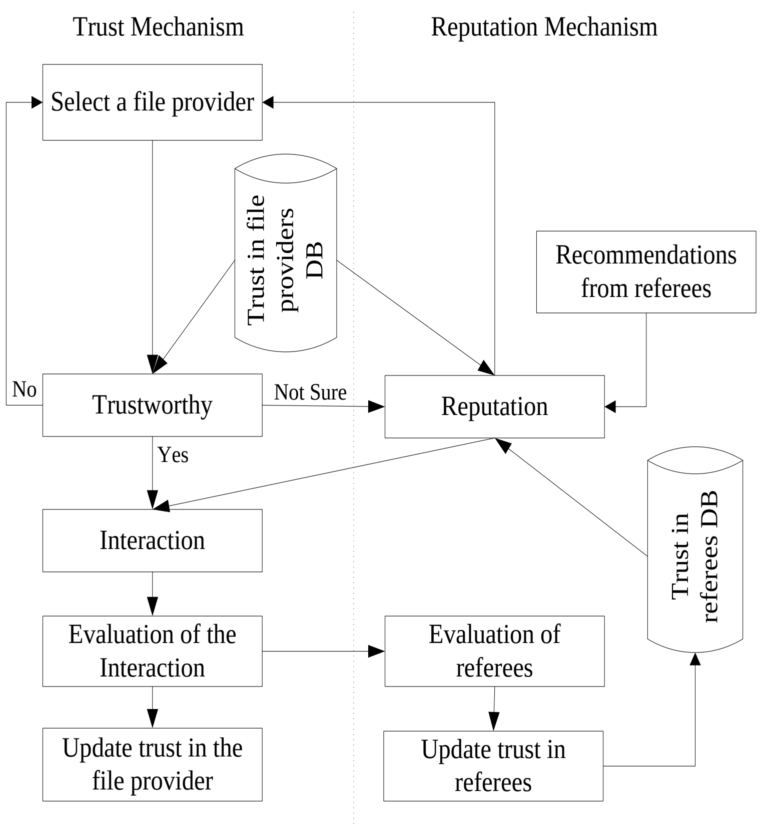


Figure 3.1 The trust and reputation mechanism

3.1.2 Trust in a File Provider's Capability

A naïve Bayesian network [25] is used to model a peer's trust in a file provider. Let's first introduce briefly naïve Bayesian networks.

3.1.2.1 A naïve Bayesian network

It is composed of a root node and several leaf nodes as Figure 3.2 shows. The root node C represents a class/goal variable with a number of values. The leaf nodes depend on the root node. These nodes represent attributes of the root node C ranging from A_1 to A_n. They are assumed to be independent given C, which means that given C, learning something about A_i tells nothing about A_j $(i \neq j)$.

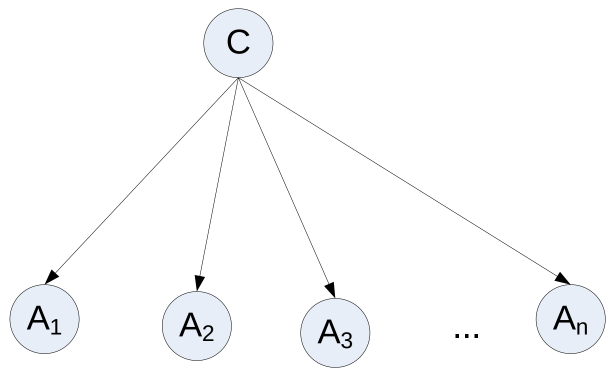


Figure 3.2. A naïve Bayesian network

The joint probability distribution (JPD) for a naïve Bayesian network can be written as follows. It is a consequence of the definition of conditional probability.

$$p(C, A_1, A_2, \dots, A_n) = p(C)P(A_1, A_2, \dots, A_n|C)$$

Since each attribute A_i is conditionally independent of any other attribute, the joint probability distribution can be rewritten as follows:

$$p(C, A_1, A_2, \dots, A_n) = p(C)p(A_1|C)p(\, A_2|C), \dots, p(A_n|C)$$

In a Bayesian network, each node is associated with a conditional probability table (CPT). The value of the joint probability in a naïve Bayesian network is the product of the values in CPTs. Therefore, a naïve Bayesian network is much faster in computation when compared with other Bayesian networks. It is also easy to understand and build.

3.1.2.2 Differentiated trust model

In a P2P network, file providers' capabilities are not uniform. For example, some file providers (FP) may be connecting through a high-speed network, while others connect through a slow modem. Some file providers might like music, so they share a lot of music files. Some may be interested in movies and share more movies. Some may be very picky about file quality, so they only keep and share files with high quality. Therefore, the file provider's capability can be represented in various aspects, such as the download speed, file quality and file type.

The peer's needs are also different in different situations. Sometimes, the peer may want to know the file provider's overall capability. Other times, it may only be interested in the file provider's capability in some particular aspect. For instance, a peer wants to download a music file from a file provider. At this time, knowing the file provider's capability in providing music files is more valuable for the peer than knowing the file provider's capability in providing movies.

Peers also need to develop differentiated trust in the file providers' capabilities. For example, the peer who wants to download a music file from the file provider cares about whether the file provider is able to provide the music file with good quality at a fast speed, which involves the file provider's capabilities in two aspects: quality and speed. How does the peer combine its two separated trust representations (e.g. the trust in the file provider's capability in providing music files with good quality and the trust in the file provider's capability in providing a fast download speed) in order to decide whether the file provider is trustworthy or not?

A naïve Bayesian network provides a flexible way to represent the trust between a peer and a file provider. Every peer develops a naive Bayesian network for each file provider that it has interacted with. Each Bayesian network has a root node T (see Figure 3.3), which has two values, "satisfying" and "dissatisfying", denoted by 1 and 0, respectively. $p(T=1)$ represents the value of peer's overall trust in the file provider's competence in providing files. It is the percentage of interactions that are satisfying. $p(T = 0)$ is the percentage of not satisfying interactions. Table 3-1 shows the conditional probability table (CPT) for the node T. a and b represent the total number of satisfying interactions and dissatisfying interactions, respectively. Table 3-1 is an extended CPT because it also stores the values of the total interaction number when $T=1$ and $T=0$, which facilitates the update for the CPT when more experiences are gained (see Section 3.1.2.3. for details).

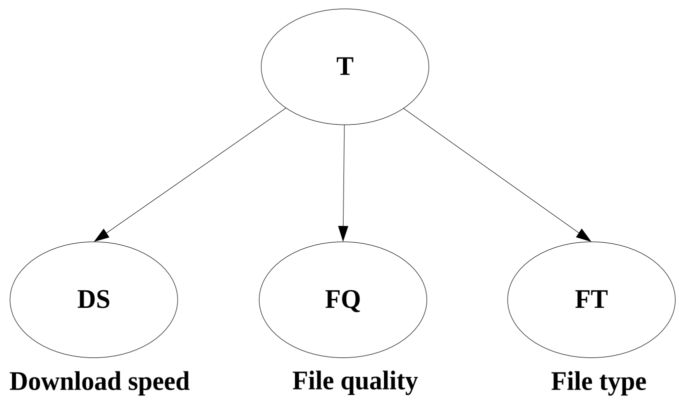


Figure 3.3 A Bayesian network model

Table 3-1 The CPT table for node "T"

Outcome	The number of interactions given the outcome	Probabilities
$T = 1$	a	$p(T=1) = a / (a + b)$
$T = 0$	b	$p(T=0) = b / (a + b)$

The leaf nodes under the root node represent the file provider's capability in different aspects. Each leaf node is associated with a conditional probability table (CPT). The node, denoted by FT, represents the set of file types. Suppose it includes five values, "*Music*", "*Movie*", "*Document*", "*Image*" and "*Software*". Its CPT is shown in Table 3-2. Each column follows one constraint, which corresponds to one value of the root node. The sum of values of each column is equal to 1. *m1, m2, ... ,m5* represent the numbers of satisfying interactions when the involved file types are "*Music*", "*Movie*", "*Document*", "*Image*" and "*Software*" respectively. Similarly, *n1, n2, ... , n5* denote the numbers of dissatisfying interactions with each file type. The reason for storing the values of *m1 to m5* and *n1 to n5* in the CPT is the same as above: to facilitate the update for the CPT when more experiences are obtained.

Table 3-2 The CPT for node "FT"

	The number of interactions given T = 1	Probabilities when T=1	The number of interactions given T = 0	Probabilities when T=0
Music	*m1*	*P(FT = "Music" | T = 1) = m1/ a*	*n1*	*P(FT = "Music" | T = 0) = n1/ b*
Movie	*m2*	*P(FT = "Movie" | T = 1) = m2/ a*	*n2*	*P(FT = "Movie" | T = 0) = n2/ b*
Document	*m3*	*P(FT = "Docu" | T = 1) = m3/ a*	*n3*	*P(FT = "Docu" | T = 0) = n3/ b*
Image	*m4*	*P(FT = "Image" | T = 1) = m4/ a*	*n4*	*P(FT = "Image" | T = 0) = n4/ b*
Software	*m5*	*P(FT = "Soft" | T = 1) = m5/a*	*n5*	*P(FT = "Soft" | T = 0) = n5/ b*

For example, *p(FT="Music"|T=1)* is the conditional probability with the condition that an interaction is satisfying. It measures the probability that the file involved in an interaction is a music file, given the interaction is satisfying. It can be computed according to the following formula:

$$p(FT = \text{Music}|T = 1) = \frac{p(FT = \text{Music}, T = 1)}{p(T = 1)}$$

$$p(FT = Music, T = 1) = \frac{the\ number\ of\ interactions\ when\ FT = Music\ and\ T = 1}{the\ number\ of\ interactions}$$

$$= \frac{m1}{a + b}$$

$$p(FT = Music|T = 1) = \frac{\frac{m1}{a + b}}{\frac{a}{a + b}} = \frac{m1}{a}$$

where *p(FT="Music", T=1)* is the probability that interactions are satisfying and files involved are music files. The other probabilities in Table 3-2 are computed in a similar way.

Node *DS* denotes the set of download speeds. It has three items, *"Fast", "Medium"* and *"Slow",* each of which covers a range of download speed.

Node *FQ* denotes the set of file qualities. It also has three items, *"High", "Medium"* and *"Low".* Its CPT is similar to the CPT for node FT in table 3.1.

Here three aspects of trust are taken into account. More relevant aspects can be added in the Bayesian network later to account for user preferences with respect to service.

Once getting nodes' CPTs in a Bayesian network, a peer can compute the probabilities that the corresponding file provider is trustworthy in different aspects, for example, $p(T = 1 \mid FT = \text{"Music"})$ – the probability that the file provider is trustworthy when providing music files, $p(T = 1, DS = \text{"Fast"}, FQ = \text{"High"})$ – the probability that the file provider is trustworthy in providing fast download speed and high quality files. The calculation is as follows:

$$p(T = 1|FT = Music) = \frac{p(T = 1, FT = Music)}{P(FT = Music)}$$

$$= \frac{p(T = 1)p(FT = Music|T = 1)}{p(T = 1, FT = Music) + p(T = 0, FT = Music)}$$

$$= \frac{p(T = 1)p(FT = Music|T = 1)}{p(T = 1)p(FT = Music|T = 1) + p(T = 0)p(FT = Music|T = 0)}$$

$$= \frac{\frac{a}{a+b} * \frac{m1}{a}}{\frac{a}{a+b} * \frac{m1}{a} + \frac{b}{a+b} * \frac{n1}{b}} = \frac{\frac{m1}{a+b}}{\frac{m1+n1}{a+b}} = \frac{m1}{m1+n1}$$

$$p(T = 1, DS = Fast, FQ = High) = p(T = 1)p(DS = Fast|T = 1)p(FQ = High|T = 1)$$

With the naïve Bayesian networks, peers can set various conditions according to their needs and infer their trust in a file provider in the various aspects from the corresponding probabilities. This will save peers much effort in building trust in each aspect separately, or developing new trust when conditions change.

3.1.2.3 Updating the Bayesian networks

A peer's trust in a file provider is built over time. After each interaction, the peer will update its corresponding Bayesian networks for the file provider to add its new experience. For example, the previous values in CPTs for node "T" and "FT" are shown in Table 3-3 and Table 3-4. After a peer downloads a music file successfully, the value of "*a*" will be increased by 1 from 8 to 9 and the value of "*m1*" from 4 to 5. Table 3-5 and Table 3-6 show how to update the CPT tables with the new experience.

Table 3-3 The CPT table for node “T”

Conditions	The number of interactions given the condition	Probabilities
T = 1	*a = 8*	*p(T=1) = a / (a + b)=0.8*
T = 0	*b = 2*	*p(T=0) = b / (a + b)=0.2*

Table 3-4 The CPT table for node “FT”

	The number of interactions given T = 1	Probabilities when *T=1*	The number of interactions given T = 0	Probabilities when *T=0*
Music	*m1 = 4*	*p(FT = “Music” | T = 1) = m1/ a = 0.5*	*b1 = 1*	*p(FT = “Music” | T = 0) = b1/ b =0.5*
Movie	*m2 = 2*	*p(FT = “Movie” | T = 1) = m2/a = 0.25*	*b2 = 1*	*p(FT = “Movie” | T = 0) = b2/ b =0.5*
Document	*m3 = 0*	*p(FT = “Docu” | T = 1) = m3/ a =0*	*b3 = 0*	*p(FT = “Docu” | T = 0) = b3/ b*
Image	*m4 = 0*	*p(FT = “Image” | T = 1) = m4/ a =0*	*b4 = 0*	*p(FT = “Image” | T = 0) = b4/ b*
Software	*m5 = 2*	*p(FT = “Soft” | T = 1) = m5/a =0.25*	*b5 = 0*	*p(FT = “Soft” | T = 0) = b5/ b*

Table 3-5 The CPT table for node “T”

Conditions	The number of interactions given the condition	Probabilities
T = 1	***a = 8+1=9***	***p(T=1) = a / (a + b)=0.82***
T = 0	*b = 2*	***p(T=0) = b / (a + b)=0.18***

Table 3-6 The CPT table for node "FT"

	The number of interactions given $T = 1$	Probabilities when T=1	The number of interactions given $T = 0$	Probabilities when T=0
Music	***m1 = 4+1=5***	***p(FT = "Music" | T = 1)= m1/ a = 5/9=0.56***	*b1 = 1*	*p(FT = "Music" | T = 0) = b1/ b*
Movie	*m2 = 2*	***p(FT = "Movie" | T = 1) = m2/ a =2/9=0.22***	*b2 = 1*	*p(FT = "Movie" | T = 0) = b2/ b*
Document	*m3 = 0*	***p(FT = "Docu" | T = 1) = m3/a =0***	*b3 = 0*	*p(FT = "Docu" | T = 0) = b3/ b*
Image	*m4 = 0*	***p(FT = "Image" | T = 1) = m4/a =0***	*b4 = 0*	*p(FT = "Image" | T = 0) = b4/ b*
Software	*m5 = 2*	***p(FT = "Soft" | T = 1) = m5/a =0.22***	*b5 = 0*	*p(FT = "Soft" | T = 0) = b5/ b*

3.1.3 Evaluating Interactions

After each interaction with the file provider, peers will evaluate the interaction. Peers might have different criteria to judge an interaction. Some peers might be very picky, but some might be generous. So they might have different evaluations of an identical interaction. The overall evaluation of an interaction is a combination of evaluations of each aspect related to the interaction, such as download speed and file quality. How to combine evaluations of each aspect depends on each peer's preference. For example, some peers may care more about the download speed. Some may care more about the quality of downloaded files. Some may equally care about both speed and quality. The result of the overall evaluation, "the interaction is satisfying" or "not satisfying", is used to update the peer's trust in the file provider. In Table 3-3, if an interaction is satisfying, the value "*a*" will be increased by 1. If it is not satisfying, the value "*b*" is increased by 1. Two main factors are considered when peers judge an interaction: the degree of their satisfaction with the download speed s_{ds} and the degree of their satisfaction with the quality of downloaded file s_{fq}. The overall degree of peers' satisfaction with an interaction s is computed as follows:

$$s = w_{ds} * s_{ds} + w_{fq} * s_{fq}\ , \quad where\ w_{ds} + w_{fq} = 1 \tag{3.1}$$

where w_{ds} and w_{fq} denote weights, which indicate the importance of download speed and the importance of file quality to a particular peer, respectively. Each peer has a satisfaction threshold s_t. If $s < s_t$, the interaction is unsatisfying; otherwise, it is satisfying.

3.1.4 Handling Recommendations

When a peer is not sure about the trustworthiness of a file provider, it can ask other peers for recommendations. The recommendation requests can vary according to a peer's needs. For example, if a peer is going to download a movie, the request can be "Is the file provider trustworthy in providing a movie file?" If the peer cares about the file quality and the download speed, the request will be something like "Is the file provider trustworthy in offering files with a good quality and a fast download speed?" When other peers receive these requests, they will check their trust representations, i.e. their Bayesian networks, to see if they can answer such questions. If a peer has downloaded movies from the file provider before, it will answer the first question with the probability $p(T = 1|FT = \text{"Music"})$ and the second question with the probability $p(T = 1, DS = \text{"Fast"}, FQ = \text{"High"})$ according to its Bayesian network.

The peer might receive several such recommendations at the same time from different references. If the references are untrustworthy, the peer can discard their recommendations immediately. Then the peer needs to combine the recommendations from the rest of references to calculate the total recommendation for the file provider as follows:

$$r_{ij} = \frac{\sum_{l=1}^{k} tr_{il} * t_{lj}}{\sum_{l=1}^{k} tr_{il}} \tag{3.2}$$

where r_{ij} is the total recommendation value for the j^{th} file provider that the i^{th} peer gets. k is the number of recommendations, respectively. tr_{il} is the trust that the i^{th} user has in the l^{th} reference. t_{lj} is the trust that the l^{th} reference has in j^{th} file provider. Given a threshold θ, if the total recommendation value is greater than θ, the peer will interact with the file provider; otherwise, not.

If the peer interacts with the file provider, it will not only update its trust in the file provider, i.e. its corresponding Bayesian network, but also update its trust in the peers that provide recommendations by the following reinforcement learning formula [82]:

$$tr_{ij}^{n} = \alpha * tr_{ij}^{o} + (1-\alpha) * e_{\alpha} \tag{3.3}$$

where tr_{ij}^{n} denotes the new trust value that the i^{th} peer has in the j^{th} reference after the update; tr_{ij}^{o} denotes the old trust value; α is the learning rate, which is a real number in the interval [0,1]; e_{α} is the new evidence value, which can be -1 or 1. If the value of recommendation is greater than θ and the interaction with the file provider afterwards is satisfying, e_{α} is equal to 1. If there is a mismatch between the recommendation and the actual experience with the file provider, the evidence is negative, so e_{α} is -1.

3.2 Experiments

In order to evaluate this approach, a simulation of a file sharing system in a P2P network was developed in Java on the JADE 2.5 (Java Agent Development Framework) [81], which is a software framework written in Java for developing agent-based applications. For the sake of simplicity, each node in the system plays only one role at a time, either the role of a file provider or the role of a peer. Every peer only knows other peers directly connected with it and a few file providers at the beginning. Figure 3.4 shows a fraction of the network. In this figure, the circles represent peers and the rectangles denote file providers.

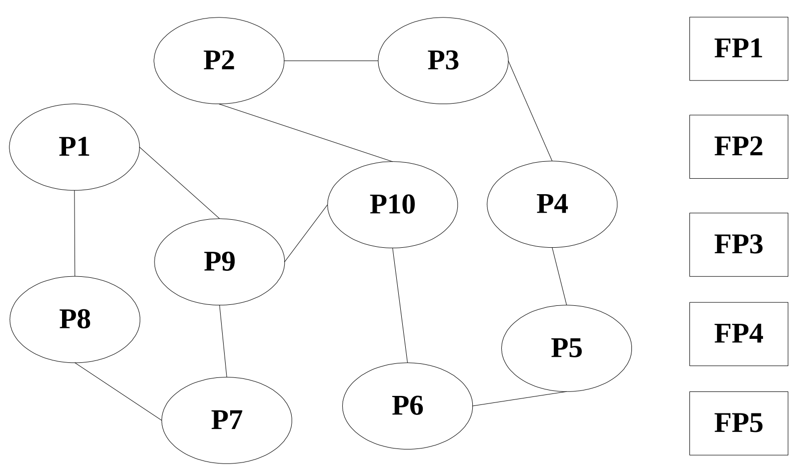


Figure 3.4 The network structure

Every peer has an interest vector. The interest vector is composed of five elements: *music, movie, image, document and software*. The value of each element indicates the strength of the peer's interest in the corresponding file type. The files the peer wants to download are generated based on its interest vector. Every peer keeps two lists. One is the peer list that records all the other peers that the peer has interacted with and its trust values in these peers. The other is the file provider list that records the known file providers and the corresponding Bayesian networks representing the peer's trust value in these file providers. Each file provider has a capability vector showing its capabilities in different aspects, i.e. providing files with different types, qualities and download speeds.

The experiments involve 10 different file providers and 40 peers. $w_{ds} = w_{fq} = 0.5$, which are the weight for download speed and file quality in formula (3.1). They are used to evaluate the satisfaction of an interaction; $\alpha = 0.9$ which is the learning rate in formula (3.3). The threshold θ is *0.5* used for judging whether a file provider or a peer is trustworthy. The total number of interactions is *1000*. Each configuration is run for *10* times and the mean is used for the evaluation criteria.

The goal of the first experiment is to see if a Bayesian network-based trust model helps peers to select file providers that match better their preferences. Therefore the system performance is measured in terms of percentage of successful recommendations. A successful recommendation is defined as a positive recommendation about a file provider such that, after receiving it and interacting with the file provider, the peer is satisfied with the interaction. The percentage of successful recommendations is the number of successful recommendations divided by the number of positive recommendations because if a peer gets a negative recommendation for a file provider, it will not interact with the file provider. So I examine the proportion of satisfactory performance over unsatisfactory performance after positive recommendation.

The performance of a system consisting of peers with Bayesian network-based trust models is compared with a system consisting of peers without Bayesian networks (BN) trust model. These peers represent general trust only, which is not differentiated into different aspects. So, there are two configurations in this experiment:

- Trust and reputation system with BN: the system consists of peers with Bayesian networks-based trust models that exchange recommendations with each other;
- Trust and reputation system without BN: the system consists of peers that do not model differentiated trust in file providers. Peers just build a general trust model in a file provider. A peer's trust in a file provider is measured by the peer's successful

interaction rate with the file provider.

Figure 3.5 shows that the performance of the system using Bayesian networks decreases at the beginning. The decrease may be caused by the learning process that peers need time to have some interactions to build Bayesian networks. After they have enough interactions for their Bayesian networks, i.e. after around 400 interactions, the system begins to perform better than the system with general trust in terms of the percentage of successful recommendations.

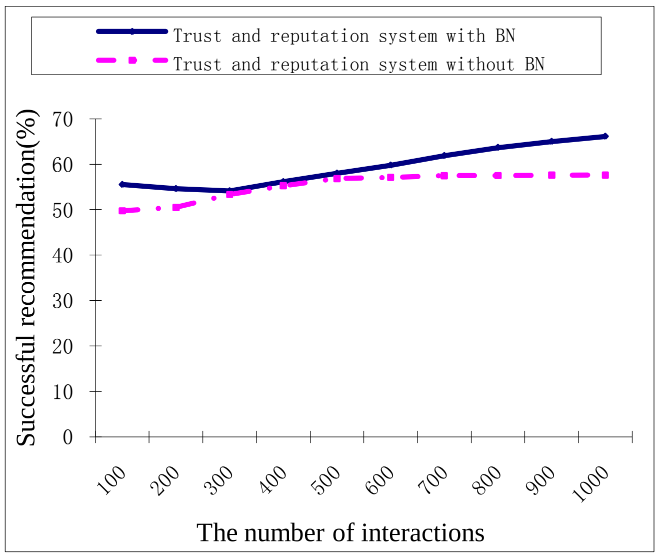


Figure 3.5 Trust and reputation system with BN vs. trust and reputation system without BN

The goal of the second experiment is to see whether peers' sharing information with each other helps achieve better performance. It is measured by the percentage of successful interactions with file providers, which is the number of successful interactions over the total number of interactions. Four configurations are compared:

- Trust and reputation system with BN;
- Trust and reputation system without BN;
- Trust system with BN: the system consists of peers with Bayesian networks-based trust models, which do not exchange recommendations with each other;
- Trust system without BN: the system consists of peers that have no differentiated trust models and do not exchange recommendations with each other.

Figure 3.6 shows that the two systems, where peers share information with each other, outperform the systems, where peers do not share information. The trust system using Bayesian networks is slightly better than the trust system without using Bayesian networks.

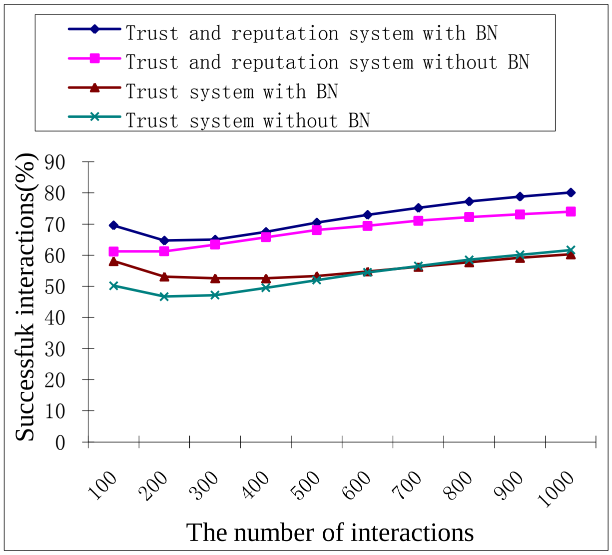


Figure 3.6 The comparison of four systems

The goal of the third experiment is to see the tendencies about successful recommendations and successful interactions in trust and reputation systems with BN. The number of interactions this time is 2000. The following three parameters are measured:

- R-Succ/Recom, the percentage of successful recommendations, which is the number of successful recommendations divided by the number of positive recommendations.
- R-Succ/Succ, the percentage of successful interactions based on recommendations, which is the number of successful interactions based on recommendations over the number of all successful interactions.
- Succ/Inter, the percentage of successful interactions in all interactions, which is the number of successful interactions divided by the total number of interactions.

Figure 3.7 shows that R-Succ/Recom and Succ/Inter tend to be stable with the increase of the number of interactions, which indicates that the percentages of successful recommendations and successful interactions are going to reach their maximal values determined by the capabilities of file providers. R-Succ/Succ tends to decrease with the increase of number of interactions, which suggests that peers need fewer and fewer recommendations when they have enough experiences with file providers.

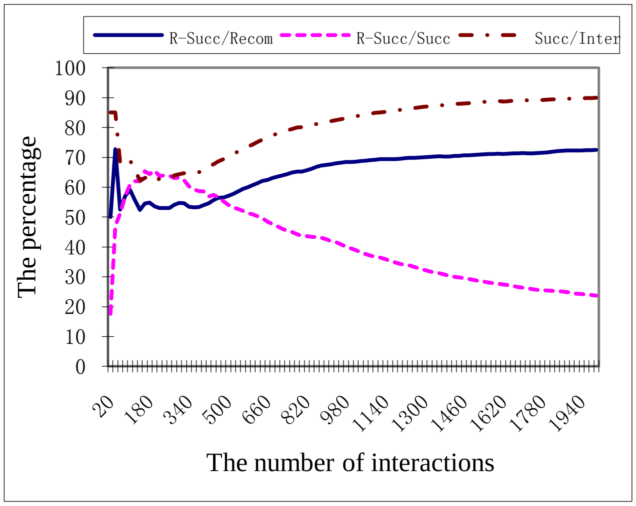


Figure 3.7 Tendency in trust and reputation systems with BN

3.3 Discussion and Related Work

Enabling peers to develop trust and reputation among themselves is important in a P2P system where resources (either computational power or files) of different quality are offered. It will become increasingly important in systems for P2P computation, where trust and reputation mechanisms can provide a way for protection from unreliable, buggy, infected or malicious peers. Trust is multifaceted. Depending on the situation, a peer may need to consider its trust in a specific aspect of another peer or in multiple aspects. Bayesian networks provide a flexible method to present the differentiated trust and combine different aspects of trust.

Although this approach is proposed for decentralized systems, the idea of using Bayesian networks to model differentiated trust can also be used for centralized systems to model differentiated reputation. For example, in eBay, Figure 3.8 shows a simple example of the Bayesian network that the central node can build to model its members' differentiated reputation. The annotation beside each node shows the values for each node. Three aspects are modeled for each eBay member. The root node "Reputation" refers to an eBay member's reputation. The node "Role" represents the role that a member could play, "As a seller" or "As a buyer". The node "Honesty" means that a member is honest or dishonest. For a seller, the honesty is to measure whether a seller is honest in providing goods as he describes in his ads. For a buyer, he is honest if he provides payment for the goods, otherwise, he is dishonest. As for the "Service Quality", a seller could provide a good service by having a good communication with his buyers or/and mailing out his goods fast. A buyer could offer a good service by making his payment without any delay.

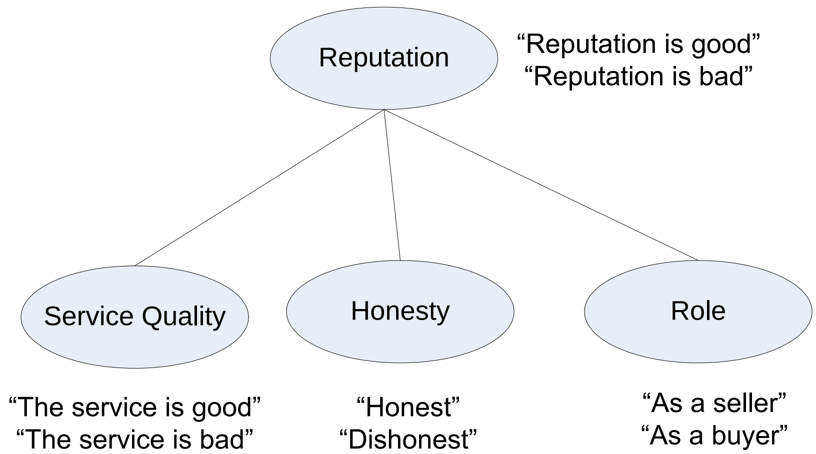


Figure 3.8 The reputation model for eBay

There is a lot of research on trust and reputation. Only the works that are most related to the approach are discussed here.

Abdul-Rahman and Hailes [3] capture the most important characteristics of trust and reputation and propose the general structure for developing trust and reputation in a distributed system. Most of the later works in the area follow their ideas, but in different application domains. Sabater and Sierra's work [53] extends the notion of trust and reputation into social and ontological dimensions. Social dimension means that the reputation of the group that an individual belongs to also influences the reputation of the individual. Ontological dimension means that the reputation of a peer is compositional. The overall reputation is obtained as a result of the combination of the peer's reputation in each aspect.

My approach integrates these two previous works [3][53]. It applies the general structure from Abdul-Rahman and Hailes [3] to develop a trust and reputation model for file sharing system in P2P networks. Like Sabater and Sierra's work, differentiated trust is modeled. Sabater and Sierra use a simple ontology to represent different aspects of trust and treats the aspects as compositional. However, the relationship between different aspects of trust is not only compositional but more complex. My model makes use of naïve Bayesian networks to represent trust in different aspects, which provides a convenient way to represent a complex relationship between different aspects of trust. My method can also tell more details about each aspect other than a general trust value. Moreover, in Sabater and Sierra's approach, the weight of each aspect has to be provided manually, which is costly and inconvenient to account for peers' different and changing needs. In my approach, peers can develop their Bayesian networks from their experiences. It provides an automatic way to adapt to peers' needs. For example, sometimes people care about the overall trust. Sometimes they only need to know the trust in some specific aspect. This bears resemblance with work on distributed user modeling and purpose-based user modeling [45][62].

Cornelli's work [15] is also in the area of file sharing in P2P networks. However, it concentrates on how to prevent attacks on the reputation system and does not discuss how peers model and compute trust and reputation.

3.4 Summary

In this chapter, a Bayesian network-based trust model is proposed to model differentiated trust and combine different aspects of trust. This model is demonstrated in a peer to peer file sharing application domain where file providers may provide different qualities of files and different speeds for downloading their files. It can be used to assist peers to find good providers to interact with. More specifically, the trust model allows peers to model the trustworthiness of file providers based on their previous interactions (prior knowledge) and takes into account the subjectivity of the peers. This model also allows peers to model the reputation of file providers based on recommendations about the providers shared by other peers. Although a P2P file sharing system is used as an example in Chapter 3, this method can be applied to other systems, like e-commerce, grid computing, web services and multi-agent systems, where differentiated trust models are needed. In order to evaluate this model, a simulation of a file sharing system in a P2P network is developed. The experiments show that the system where peers communicate their experiences (recommendations) outperforms the system where peers do not communicate with each other and that a differentiated trust adds to the performance in terms of percentage of successful interactions.

Chapter 4

The Framework for Super-Agent Based Reputation Management

In Chapter 3, a method of using Bayesian networks was proposed to model differentiated trust that can automatically adapt to agents' different needs. Although the Bayesian network based method provides a good way to model differentiated trust, it is still difficult and inefficient for agents to collect information to build reputation in decentralized systems.

Compared with reputation approaches in centralized systems, decentralized reputation management approaches are quite different. In a centralized system, like eBay, a central entity takes all the responsibilities to collect and provide reputation information for its members. The central entity is regarded as an authority. It can store all the information and is assumed to be truthful and available whenever a member wants to get information from it. However, this is not the case in decentralized systems, where agents have to cooperate to share information so that they can collect information and build reputation for each other. For instance, in Yu and Singh's approach [86], in order to know the reputation of an agent A, an agent B has to seek many other agents' opinions about A and then combine their opinions together. Nevertheless, during this process, the agent B may not be able to find its required information. One possible reason is that agent B cannot find the agents that have the information since it does not know which agents have interacted with the agent A and can provide the information. When it sends its search requests blindly, they may not reach these agents that have the information. The other reason could be that the agents that have the information cannot provide it at the moment of request because of their poor capabilities in terms of CPU power, bandwidth, and availability. For example, they may happen to be offline, or too busy and run out of CPU power or bandwidth. Their opinions will not be considered when agent B builds agent A's reputation. Therefore, agent A's reputation may be unreliable to use for making a decision.

To deal with this problem, a super-agent based reputation mechanism is proposed in this chapter for decentralized systems. The idea is to make good use of the resources of super-agents which are agents with more resources (i.e. extra CPU power, larger storage and higher network bandwidth) [70]. More specifically, super-agents take additional responsibilities: collecting and storing feedback about services, building reputation of services, and sharing the reputation information with other agents (e.g. the agents with poor capabilities) in the system. Consumer agents carefully select which super-agents' advice to follow by modeling the trustworthiness of these super-agents. The consumers that benefit from super-agents also provide the super-agents with feedback about their interactions with service providers after evaluating the interaction results. In order for my super-agent based reputation management to work effectively, super-agents have to contribute resources to maintain reputation information and answer queries about reputation of services. These super-agents may be malicious in providing reputation information. They may provide false good reputation for some services to promote them or provide false bad reputation to bad-mouth some other services. To deal with this problem, I have designed a practical reward mechanism, inspired by real world examples where service providers offer rewards for agents that bring consumers to consume their services. Super-agents that are honest and contribute more resources will attract a larger number of consumers to follow their advice about services. These super-agents will then be able to obtain more rewards from the service providers. A service selection environment is simulated, where some service providers may provide low quality of services and some super-agents may be malicious. The experimental results confirm that by using the proposed super-agent based approach, consumer agents are more likely to find good services with lower searching cost, compared with the approaches that do not use super-agents including the model of Yu and Singh [3]. Service providers are better off offering rewards to super-agents, in order to attract more consumers. And, super-agents are incentivized to contribute more resources and be honest, in order to gain more rewards.

The idea of using super-agents for reputation management is inspired from super-peer networks, which are more efficient than pure P2P networks in terms of searching resources and passing messages. Super-peers are peers with more capabilities. Peers with poor capabilities are connected to super-peers. A super-peer acts as a server for a small group of clients (i.e. peers with poor resources) to store their information, and to send and receive messages for them. In this way, peers with poor capabilities will not cause system bottlenecks like they do in pure P2P networks. My idea of using super-agents for reputation management also takes advantage of the extra power of super-agents. It is novel since other existing trust and reputation mechanisms in decentralized systems do not pay attention to the role of super-agents.

The rest of this chapter is organized as follows. Section 4.1 gives the details about this mechanism. Sections 4.2 introduces the reward mechanism to reward super-agents for contributing their resources and being honest. The experimental design and results are presented in Section 4.3. Section 4.4 discusses related work. A summary is presented in the last section.

4.1 Super-Agent Based Reputation Management

In this section, first, an overview of the super-agent based reputation management framework is provided. Then the formalization of each modeling process is described.

4.1.1 Overview of the Framework

The super-agent based mechanism is general and applicable to different decentralized systems, like P2P systems, e-commerce, web services and multi-agent systems. For generality, agents are assumed to provide and consume services. The services could be web services, file sharing, or selling/buying goods. The agent that provides services is called "*service provider*". The agent that consumes services is called "*consumer agent*" or simply "*agent*". The reputation mechanism described in this chapter is an enhancement to the Bayesian network based mechanism described in the previous chapter, with the involvement of super-agents.

4.1.1.1 The role of super-agents

In the framework, super-agents are responsible for collecting information, building reputation for services, and providing the services' reputation information to agents. Agents send their feedback to super-agents. Super-agents are independent. They decide for which services that they want to build reputation. These could be services that they are interested in or services that they want to consume in the future. Some services may have multiple super-agents building reputation for them, while some services may not have any. In this case, agents can still ask other normal agents (non super-agents) for information to build service reputation using the mechanism presented in Chapter 3.

An agent can discover super-agents through search requests. When an agent searches for a service, it sends a search query. When a super-agent receives a query about a service for which it is building reputation, it sends a reply message to the service requestor to let it know about its existence. The drawback of this method is that there is no guarantee of finding super-agents that built reputation for the service.

An alternative way is that an agent can contact the agent that provides the service, i.e. the service provider. The service provider maintains and provides a contact list for each of its services. A contact list of a service is a list of super-agents that are building reputation for the service. When a super-agent wants to build reputation for a service, it tells the service provider to add it into the service's contact list, which contains all super-agents that the service provider knows that build reputation for the service. The potential problem with this method is that a service provider may not like to put super-agents providing a negative reputation into its service's contact list. If this is the case, the previous method of discovering super-agents through search requests can still help agents to find the super-agents that are not suggested by the service provider.

When an agent gets a list of super-agents that are building reputation for a service, it will judge the trustworthiness of each of these super-agents to decide whether to use the reputation information it provides. The trustworthiness of a super-agent measures the reliability of the reputation information it provides, which implies that a super-agent is honest and the information it provides is correct. If there is no trustworthy super-agent, an agent will build the service's reputation using the mechanism in Chapter 3 by collecting information from other normal agents (non super-agents). In order to help super-agents collect information and build reputation, after agents use a service, they can send their feedbacks about the service to the super-agents they find.

The next section explains how agents select services and how super-agents are involved in the process.

4.1.1.2 Procedure for an agent selecting a service

Figure 4.1 shows the procedure of how an agent selects a service. When an agent wants to find a service, it issues a search query using keywords. If a service provider receives the query, it checks whether it provides the required service. If yes, it returns a message to the querying agent about the service, including the service name and the description of its service. When the querying agent receives the return message, it adds the service into its service's list, which is used to store the service information found from a search query. If a super-agent receives the search query, it checks whether it is building reputation for a service matching the search keywords. If yes, it sends a message containing the service's information (e.g. the service name and the description of its service) to the querying agent and also tells it that it is building reputation for the service. When the querying agent receives the return message from a super-agent, it not only adds the service into its service list, but also sets up a super-agent list for the service to store the information about the super-agents in case it will need reputation information from them later. In this way, super-agents not only help service providers to propagate their services' information so that more agents can learn their services, but also let agents know them. As mentioned before, super-agents can be found through search queries. So even when a super-agent is not listed in the service's contact list by a service provider, agents still have a chance to find it through the searching process.

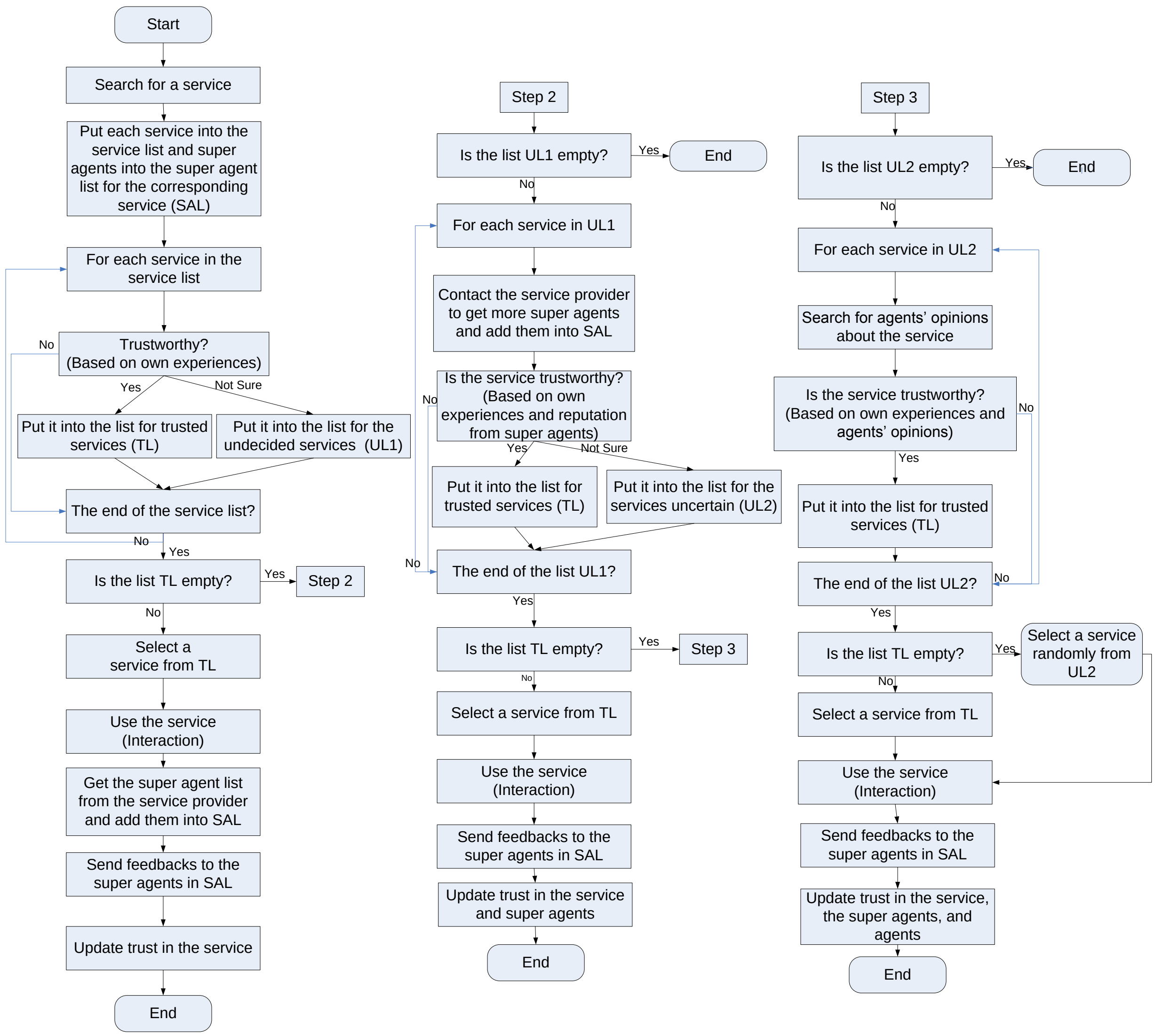


Figure 4.1 The steps for selecting a service

After an agent gets a list of services that match its search request, it has to select a service that is the most trustworthy to use. The trustworthiness of a service is used to measure whether it can meet an agent's expectation in terms of the quality of the service measured by multiple metrics.

These metrics are different in different systems. For a web service system, it could be a service's response time or execution time, or the accuracy of a computing result. For a file sharing system, it could be the download speed and/or file quality.

An agent's decision about a service's trustworthiness involves three steps. In the first step, the agent uses its own experiences to find trustworthy services from its service's list without asking super-agents or other agents. If an agent has used a service many times and has enough experiences with it, it can decide the trustworthiness of the service from its own experiences. If a service is trustworthy, it puts it into a list TL created for trusted services. If a service is not trustworthy, it just discards it. If an agent has not enough experiences to decide the trustworthiness of a service, it puts it into the list UL1 created for undecided services. After examining the services in the service list, the agent gets two lists. One is TL for trusted services and another one is UL1 for undecided services. If TL is not empty, which means that the agent has found some trustworthy services based on its own experiences, it can select to use one of them. However, if TL is empty, the agent has to go to the step 2 and get help from super-agents to find out the trustworthiness of the services in the list UL1, if it is not empty. For each service in UL1, the agent will contact the service's provider to see whether it can provide the service's contact list containing the super-agents building reputation for the service.

The agent can merge its list of super-agents that it has found through its search query with the list from the service provider to get a final list of super-agents. Then the agent selects the trustworthy super-agents from the list to ask for the service's reputation. The agent combines its own experiences with the service's reputation to judge the trustworthiness of the service. If it is trustworthy, the agent puts it into the list TL. If it is not trustworthy, the agent discards it. If the agent can't find super-agents for a service or does not trust these super-agents, it cannot decide about the service's trustworthiness. In that case, the agent will put the service in a new list created for the undecided services (UL2). After all the services have been examined in UL1, the agent still gets two lists, the list of trusted services TL and the list of undecided services UL2. If TL is not empty, it means that the agent has found some trustworthy services based on the combination of its own experiences and the reputation of the services. Then the agent can choose a service in TL to use.

If TL is still empty after step 2, the agent will go to the step 3. It has to send queries to other normal agents and ask for their opinions about the trustworthiness of the services in UL2. Then it can combine together its own experiences and the opinions from other normal agents to decide the service's trustworthiness. If it is trustworthy, it is put into the list TL. If TL is not empty, the agent can select a service to use. After using the service, the agent will send its feedback about the service to the super-agents that it has found, so that they can update their information about the service's reputation.

4.1.1.3 Security considerations

The proposed approach raises two security considerations.

The integrity of the data. Messages can be easily tampered with or modified during storage or transmission. There are a number of known security threats in P2P networks, such as the attack of "man in the middle". In order to guarantee the integrity of data, a PKI-based scheme can be used, which requires each agent (a super-agent or an agent) to have a public and private key pair. The agent ID will be either a digest of its public key obtained using a secure hash function, or the public key itself. The basic idea is that when an agent creates a message, such as a search request or a feedback, it will encrypt the message using its private key and send the message along with its public key. When an agent receives an encrypted message, the public key coming along with the message can be used to decrypt the message.

Only the agent that has used a service can submit its feedback to super-agents. When an agent uses a service, the service provider will issue a certificate to the agent encrypted with its private key as a proof that the peer has used the service. When the peer submits its feedback along with the certificate to super-agents, super-agents can use the service provider's public key to decrypt the certificate. If the data is corrupted, which means that the certificate is invalid, super-agents will discard the feedback, otherwise they will accept it.

4.1.2 Bayesian Network Based Trust/Reputation Model of Services

The approach of the Bayesian network based trust modeling described in Chapter 3 will be used for a consumer agent to model trust in a service based on its own experiences. It will also be used for super-agents to build reputation for services.

A consumer agent can build a differentiated trust model about a service based on its own experiences. Figure 4.2 shows the example of the Bayesian network based trust model for a service that provides music file downloading. A consumer agent can evaluate the service based on the downloading speed and the file quality. The overall evaluation of an interaction between a consumer and a service is a combination of the evaluation for two aspects. How to combine the evaluations of the two aspects depends on the agent's own judging criteria. For example, some agents may consider the downloading speed more important. Others may care more about the quality of the music file. The result of the overall evaluation about an interaction with the service is either "satisfying" or "not satisfying", which is used to update the consumer agent's trust in the service after the interaction.

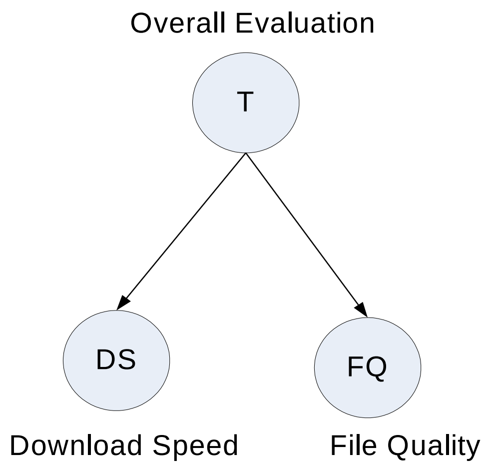


Figure 4.2 The Bayesian network based trust model of a file downloading service

Two main factors are considered when an agent judges an interaction, the degree of their satisfaction with the download speed s_{ds} and the degree of their satisfaction with the quality of downloaded file s_{fq}. The overall degree of an agent's satisfaction with an interaction s is computed as the following:

$$s = w_{ds} * s_{ds} + w_{fq} * s_{fq} \quad where\, w_{ds} + w_{fq} = 1 \tag{4.1}$$

where w_{ds} and w_{fq} denote weights, which indicate the importance of download speed and the importance of file quality to a particular agent, respectively. Each agent has a satisfaction threshold s_t. If $s < s_t$, the interaction is unsatisfying; otherwise, it is satisfying.

Figure 4.2 shows the Bayesian network has a root node T, which has two values, "satisfying" and "unsatisfying", denoted by 1 and 0, respectively. *p(T=1)* represents the value of the peer's overall trust in the file downloading service. It is the percentage of interactions that are satisfying.

p(T = 0) is the percentage of not satisfying interactions. Table 4-1 shows the conditional probability table (CPT) for the node T. *a* and *b* represent the total number of satisfying interactions and dissatisfying interactions, respectively. It is an extended CPT because it also stores the values of the total interaction number when *T=1* and *T=0*, which facilitates the update for the CPT when more experiences are gained.

Table 4-1 The CPT table for node "T"

Outcome	The number of interactions given the outcome	Probabilities
$T = 1$	a	$P(T=1) = a / (a + b)$
$T = 0$	b	$P(T=0) = b / (a + b)$

The leaf nodes under the root node represent the different aspects of the service. Each leaf node is associated with a conditional probability table (CPT).

Table 4-2 shows the CPT for node "*DS*", which denotes the set of download speeds. It has three items, "*Fast*", "*Medium*" and "*Slow*", each of which covers a range of download speeds. Each column follows one constraint, which corresponds to one value of the root node. The sum of values of each column is equal to 1. *d1*, *d2* and *d3* represent the numbers of satisfying interactions when the download speed is "*Fast*", "*Medium*" and "*Slow*", respectively. Similarly, *f1*, *f2* and *f3* denote the numbers of dissatisfying interactions for each value of node *DS*. Node *FQ* denotes the set of file qualities. Its CPT is similar to that of the node *DS*. It also has three items, "*High*", "*Medium*" and "*Low*".

Table 4-2 The CPT for node "DS"

	The number of interactions given T = 1	Probabilities when T=1	The number of interactions given T = 0	Probabilities when T=0
Fast	*d1*	*P(FT = "Fast" | T = 1) = d1/ a*	*f1*	*P(FT = "Fast" | T = 0) = f1/ b*
Medium	*d2*	*P(FT = " Medium " | T = 1) = d2/ a*	*f2*	*P(FT = " Medium " | T = 0) = f2/ b*
Slow	*d3*	*P(FT = " Slow " | T = 1) = d3/ a*	*f3*	*P(FT = " Slow " | T = 0) = f3/ b*

After each interaction, an agent will update all the CPTs in its trust model of a service according to the method described in Section 3.1.2.3.

A super-agent will also use Bayesian network based trust modeling to build reputation for a particular service. The way it builds reputation for a service is the same as if it builds trust from its own experiences. The super-agent just needs to treat the feedbacks from different consumer agents as its own experience when building reputation for the service.

4.1.3 The Trustworthiness of a Service

When a consumer agent *c* judges the trustworthiness of a service *s*, it will first use its own experiences. If it does not have enough personal experience with the service *s*, it will ask super-agents for information about the service. For the same service, there may be multiple super-agents building reputation for it. They are independent and build the service's reputation based on the feedbacks they received. Since an agent may not be able to find all the super-agents that build reputation about a service and send feedbacks to all of them, some super-agents may receive more feedbacks and some may receive less. Therefore, they may have different reputation values for the same service. Discrepancies could also be caused by other reasons. For example, a super-agent may miss a couple of feedbacks because of occasional disconnection. Or a super-agent could be dishonest and want to bad mouth or boost up a service. Therefore, an agent has to ask multiple super-agents and weigh their information based on its trust in them to get a collective view of a service's reputation. The agent sorts the list of super-agents according to its trust in them from high to low. If the agent's trust in a super-agent is higher than a threshold, the super-agent will be regarded as trustworthy and will be asked for advice about the service. A super-agent which is asked for advice will provide a reputation opinion about the service, a value in the interval [0, 1] where 0 means that the service is totally disreputable and 1 means that the service is completely reputable. Once the consumer agent receives all reputation opinions about the service from all trustworthy super-agents $\{sp_1, sp_2, ..., sp_k\}$, the aggregated reputation value can be calculated according to the following weighted average formula:

$$R_{sp}(s) = \frac{\sum_{i=1}^{k} T(sp_i) * R(sp_i, s)}{\sum_{i=1}^{k} T(sp_i)} \tag{4.2}$$

$R_{sp}(s)$ denotes the service's collective reputation from super-agents. k is the number of super-agents. $T(sp_i)$ is the agent's trust in the *i-th* super-agent . The modeling of the trustworthiness of super-agents will be described in the next section 4.1.4. $R(sp_i, s)$ is the reputation of the service s from the *i-th* super-agent. A super-agent uses the Bayesian network based method to model the differentiated reputation of a service. It can get different values for $R(sp_i, s)$ from its Bayesian network depending on the querying agent's needs. If the querying agent cares about the service's overall performance, the super-agent would use the percentage of the satisfying interactions. If the querying agent cares about some specific aspect of the service, it can get a value calculated based on the querying agent's requirement. For simplicity, in my simulation, I assume that agents only care about a service's overall performance. Therefore, the percentage of the satisfying interactions will be used as the reputation of the service built by a super-agent.

The trustworthiness of service s $T(s)$ is determined by both the consumer agent's personal experience and the opinions provided by super-agents. It can be calculated based on the combination of the consumer agent's trust $T'(s)$ in the service based on its own experience and the aggregated reputation opinion $R_{sp}(s)$ of super-agents, as follows:

$$T(s) = w * T'(s) + (1 - w) * R_{sp}(s) \tag{4.3}$$

T'(s) is a representation of an agent's past experiences and shows the extent of trustworthiness that an agent has in the service based on its own experiences. It is modeled using the Bayesian network-based method. An agent can get different values for *T'(s)* from its Bayesian network depending on its requirements. For simplicity, in my simulation, the percentage of the satisfying interactions will be used as the trust value that an agent has of a service.

In formula (4.3), *w* represents how much weight should be put on *T(s)*. It is determined based on the number of interactions between the agent and the service *s*. More specifically, it is determined by the minimum number of interactions needed for an agent to be confident about the trust value it has of *s*, computed based on the agent's personal experience. Based on the Chernoff Bound theorem [90], the minimum number of interactions N_{min} can be determined by an acceptable level of error ε and a confidence measurement γ:

$$N_{min} = -\frac{1}{2\varepsilon^2} In \frac{1-\gamma}{2} \tag{4.4}$$

If the total number of interactions N_{all} is larger than or equal to N_{min}, the agent has enough personal experience with the service and will be confident about the trust value estimated based on its personal experience. Otherwise, the agent will also consider the aggregated reputation $R_{sp}(s)$ of the service calculated based on reputation opinions provided by super-agents. The weigh *w* can be measured as follows:

$$w = \begin{cases} \frac{N_{all}}{N_{min}}, & if\ N_{all} < N_{min}; \\ 1, & otherwise \end{cases} \tag{4.5}$$

When *w* equals 1, the trustworthiness of the service is the same as the trust value calculated based on only the agent's personal experience with the service (see Equation 4.3). When *w* is less than 1, the aggregated reputation $R_{sp}(s)$ of the service also plays a role in the calculation of the trustworthiness of the service.

Note that there may be the case where a consumer agent does not have enough experience with a service and it also cannot find trustworthy super-agents to ask for reputation opinions about the service. In this case, the consumer agent will also ask advice about the service from other consumer agents which may have interacted with the service before. The calculation of an aggregated reputation value based on other consumer agents' advice and the equation for combining the consumer agent's own experience with the aggregated reputation value are also similar to Equations 4.2 and 4.3.

4.1.4 The Trustworthiness of a Super-Agent

When an agent asks a super-agent about a service's reputation, it can develop trust in the super-agent based on its experience of using the service. A service's reputation from a super-agent $R(sp_i, s)$ can be represented as a value between [0, 1], calculated from the super-agent's Bayesian network according to the querying agent's requirement. "*1*" denotes a good reputation and "*0*" means a bad reputation. After using a service, an agent will evaluate this experience. The evaluation can be "satisfying" or "dissatisfying". The reinforcement learning formula is used to model the trustworthiness of a super-agent, as follows:

$$T(sp_i) = \alpha T'(sp_i) + (1 - \alpha)e(sp_i) \tag{4.6}$$

where $T(sp_i)$ denotes the agent's trust in the super-agent sp_i after the update; $T'(sp_i)$ denotes the trust value before the update. α is the learning rate, a real number in the interval [0,1]. $e(sp_i)$ is the new experience and represents the agent's evaluation about sp_i, a value between 0 and 1. "*0*" means a bad evaluation and that the reputation the super-agent provides is false. "*1*" is a good evaluation and the super-agent provides the right information.

$$e(sp_i) = \begin{cases} R(sp_i, s), & if\ e(s) = 1 \\ 1 - R(sp_i, s), & if\ e(s) = 0 \end{cases} \tag{4.7}$$

The value of $e(sp_i)$ is determined by comparing an agent's own experience of using a service with the service's reputation that the super-agent provides. If the agent's experience of using a service is satisfying, $e(sp_i)$ is equal to the service's reputation value from the super-agent, i.e. $R(sp_i, s)$. If the agent's experience of using a service is dissatisfying, $e(sp_i)$ equals $(1- R(sp_i, s))$. For example, if a super-agent says that a service's reputation is 0.9 and the agent's experience is satisfying, the service's reputation is consistent with the agent's experience. Therefore, $e(sp_i)$ equals 0.9. However if the service's reputation is 0.9 and the agent's experience is dissatisfying, it indicates there is a mismatch. Therefore, $e(sp_i)$ equals 0.1 (i.e. 1- 0.9). A super-agent can gain more trust if the reputation value it provides matches the agent's experience more.

In the case where a consumer agent does not have enough experience with a service and it also cannot find trustworthy super-agents to ask for reputation opinions about the service, the consumer agent will ask advice about the service from other consumer agents which may have interacted with the service before. It will also develop trust in other consumer agents based on its experience of using the service. The trustworthiness of a consumer agent is modeled in the same way as that of a super-agent. The calculation of the trustworthiness value is also similar to Equations 4.6.

4.2 Reward mechanism

In the super-agent based reputation management, super-agents have to contribute more resources to collect, store and maintain reputation information about services and to answer queries of consumer agents. Super-agents need incentives for contributing resources and sharing information with consumers. In addition, some super-agents may be dishonest in providing reputation information. They may provide false good reputation for some services to promote these services or provide false bad reputation to bad mouth some other services. To address these two problems, I design a reward mechanism to create incentives for super-agents to contribute resources and share truthful reputation information about services. Inspired by real world examples, the reward mechanism is designed to be rather simple but practical.

More specifically, in the reward mechanism, service providers will provide rewards to super-agents. Each service provider can issue its own "virtual points", similar to "store credits" in the real world. When a customer accumulates enough "store credits", these credits can be redeemed for goods in the store. When a consumer agent consumes a service provided by a service provider and is also satisfied with the service, the consumer agent will tell the provider a list of super-agents from whom the agent selects to ask for advice about the service. A number of "virtual points" will be awarded to these super-agents. However, if the consumer agent is not satisfied with the service after consuming it, it will not provide its list of super-agents to the service's provider. Therefore, no super-agents will be rewarded. This will prevent super-agents from gaining rewards by providing fake good reputation of services. The number of "virtual points" may be dependent on the value of the service consumed by the consumer agent and the total number of super-agents reported by the consumer agent. To keep the reward mechanism simple, I assume that the "virtual points" will be equally distributed among these super-agents. This simplification is reasonable because the total number of super-agents providing advice to a consumer agent about a service is not expected to be large. The simplification has also often been applied in the real world. The "virtual points" issued by a service provider can be used to redeem services offered by this provider. These "virtual points" may also be used to provide super-agents higher priorities to consume services or provide them with higher quality of services. Service providers have obvious incentives to provide rewards to super-agents. Super-agents building reputation for services offered by the service providers will help the service providers propagate their service information and therefore potentially bring them more consumers.

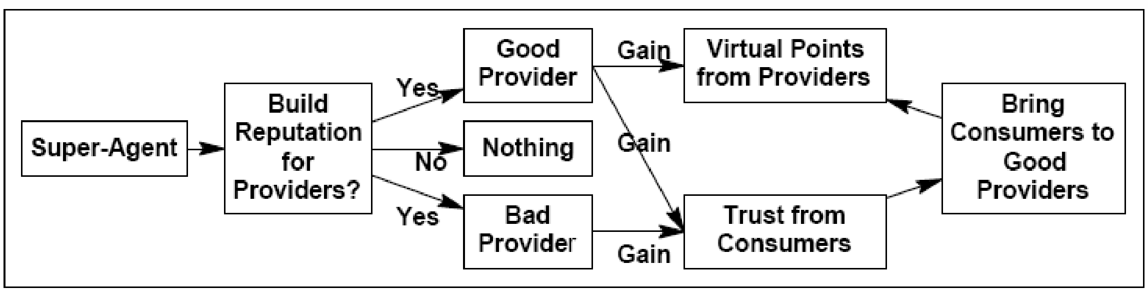


Figure 4.3 Incentives for building reputation for good and bad providers

For super-agents, if they build reputation for good services, they can gain "virtual points" from the providers of these good services (see Figure 4.3). The super-agents can then redeem the points for their future interactions with the service providers, i.e. consuming the good services. If some services are bad, super-agents may not gain "virtual points" from the providers of these services because consumer agents will likely not consume these services. But, it is still beneficial for super-agents to build reputation for bad services. They gain trust from consumers by reporting honestly the bad service's reputation. This can increase the super-agents' chance of being asked for advice by the consumer agents and the ability to gain points from good service's providers. Generally speaking, if a super-agent contributes more resources to maintain reputation information about services, and truthfully shares the reputation information with consumer agents, it will be trusted by many consumer agents. It is then able to bring more consumer agents to consume good services. Their good behavior will be rewarded by the service providers providing these good services with virtual credits that the super-agent can use itself to consume the good services for which it builds reputation.

4.3 Experiments

In this section, experiments are carried out to compare the effectiveness and scalability of my approach with one that does not use super-agents and with the approach proposed in Yu and Singh's model [86]. I also carry out experiments to demonstrate the strong incentives created by my approach for service providers to reward super-agents and for super-agents to contribute resources and provide truthful reputation information about services.

A service selection environment is simulated and written in C# and run on a PC with Intel(R) Xeon(TM) CPU 3.20 GHz and 2.0G RAM. It involves service providers and consumers, some of which are super-agents. Consumer agents and super-agents both consume services provided by service providers. A matrix with 5×5 cells is used to simulate a peer to peer (P2P) system as shown in Figure 4.4. The accessibility of peers in P2P environments is mapped to the matrix. Agents in the same cell are neighboring peers that can reach and communicate with each other by one or more hops. Originally, service providers (shown as stars), consumers (shown as white circles) and super-agents (shown as black circles) are randomly located in the cells. Consumers and super-agents are different in their ability in discovering service providers. Consumers can only find directly the service providers in their own cell. Super-agents are able to directly find the service providers not only in their own cells, but also in the cells adjacent to their own cells. For example, in Figure 4.4, consumer *C1* can find directly provider *P1* but not *P2*. Super-agent *S1* can directly find both *P1* and *P2*. This simulates that super-agents have more searching power than ordinary consumers in the network. Super-agents build reputation for services provided by the service providers within their searching scope. Thus, one service provided by a service provider may have several super-agents build reputation for it. For example, in the figure, super-agents *S2*, *S3* and *S4* all build reputation for a service provided by *P3*. In the simulation, super-agents also connect with the consumers in their own cells as well as the cells adjacent to them. In this way, consumers are able to find through super-agents the service providers that are not in the consumers' own cells. For example, *C1* can find *P2* through super-agent *S1*.

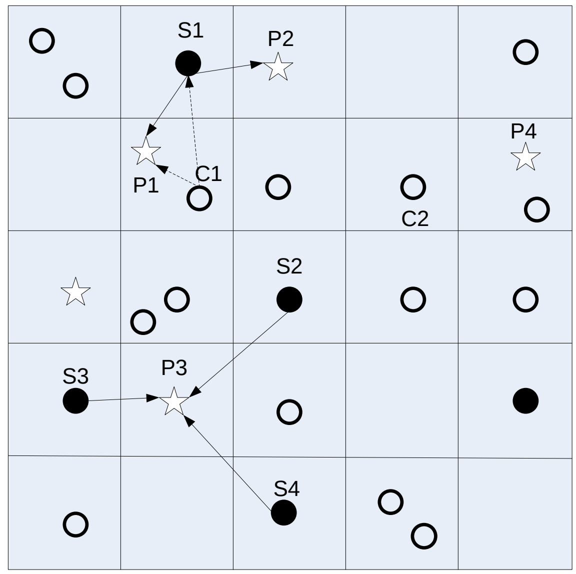


Figure 4.4 An example of a simulated service selection environment

Since the purpose of this simulation is to show the benefits of using super-agents, not the benefits of using Bayesian network based trust modeling, which has been demonstrated in the experiments of Chapter 3, I can simplify the simulation. After a consumer agent uses a service, it will only gives an overall evaluation about the interaction, which is either as “satisfying” or “dissatisfying”. Therefore, when a consumer agent models its trust in a service provider using the Bayesian network based method, it has no leaf nodes, but a root node, which represents its overall evaluation about the service. The probability of successful interactions will be used as the trust value that a consumer agent has of the service.

There are three types of service providers in my simulation: superior service providers, normal service providers and bad service providers. They are different in their ability to provide satisfying services. The superior service provider has 90% probability to provide satisfying services. The normal service provider has 60% probability, and the bad one has only 40% probability. The numbers of providers of each type are 1, 3, and 1 respectively. For the purpose of simplicity, all the services offered by the providers have the same functionality. Each service provider provides one service. The experiments involve 100 consumer agents. super-agents should have both a large bandwidth and a long availability. According to studies [55] of Napster and Gnutella, which are two popular file sharing systems, only 20% users have a large bandwidth or a long availability. It can be assumed that the percentage of users with both a large bandwidth and a long availability is less than 20%. To be a little pessimistic, I assume 10% of consumer agents are super-agents in the experiments. There are 4000 interactions in each experiment. In each interaction, a consumer agent selects a service provider and uses its service. I run each experiment for 10 times and present the average of the results produced by each experiment. The values for α, γ, ε are *0.9*, *0.7*, and *0.3*, respectively.

4.3.1 Demonstrating Benefits of Using Super-Agents

In the first set of experiments, I compared my super-agent based approach with the other two approaches. One approach does not use super-agents, and consumers themselves ask other consumers for reputation opinions about services. The other approach is that of Yu and Singh described in Section 2.3.3. The difference between these two approaches is that Yu and Singh's approach also allows a consumer to ask another consumer for referrals – other consumers that can provide reputation information about services, and in this way, more information about the services may be obtained by the asking consumer. In this experiment, I measured the effectiveness of an approach based on the ratio of successful interactions. A successful interaction means that a consumer agent selects a service provider, uses its service, and finds it satisfying. Figure 4.5 shows that the super-agent based approach performs much better than the other two approaches. Yu and Singh's approach performs slightly better than the approach that does not use super-agents. These results confirm that super-agents in my approach can more effectively help consumers find potentially good service providers.

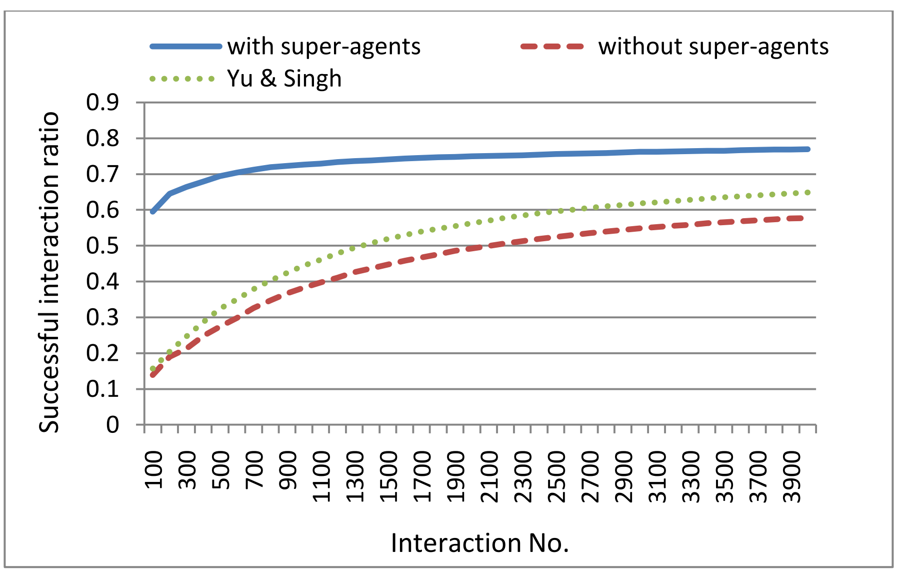


Figure 4.5 Comparison of successful interaction ratio

The second experiment involves some percentage of malicious super-agents, varying from 20% to 100%. They provide untruthful reputation information about services. The results plotted in Figure 4.6 shows that only when all super-agents are malicious, my approach is slightly worse than the approach that does not use super-agents. In my approach, consumers effectively model the trustworthiness of super-agents, and select the trustworthy ones for advice about services, based on which they can select satisfying services.

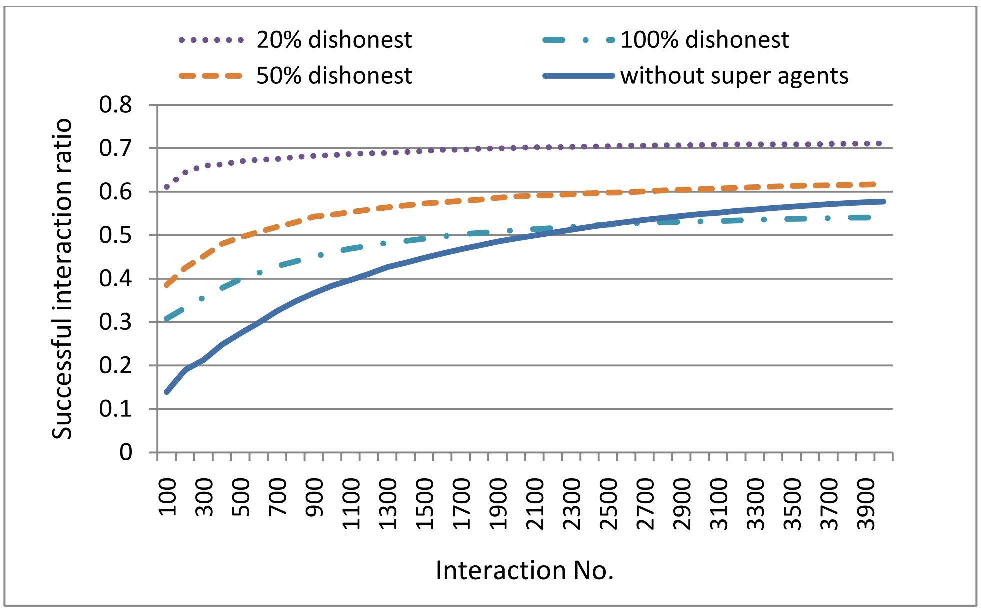


Figure 4.6 Dishonest super-agents versus no super-agents

In my simulation, I allow consumers to move from a cell to a neighbor cell for finding a service provider to interact with. For example, as shown in Figure 4.7, consumer *C2* cannot directly see service provider P4. But, it can move from its own cell to the neighbor cell where *P4* is located, demonstrated as the dashed circle *C2*. It now can find *P4* in the new cell. In this way, the searching cost (scalability) of a system can be measured as the total number of moves that consumers have taken for finding good service providers in all interactions. A lower searching cost implies that it takes less time for consumers to find good services and the number of messages passed between consumers is smaller. In this experiment, I compare the searching cost of the system that uses my approach with those that use the other two approaches. As shown in Figure 4.8, the consumers in the system with my approach need a much smaller number of moves in order to find good services. The system that uses Yu and Singh's approach is slightly better than the system without super-agents.

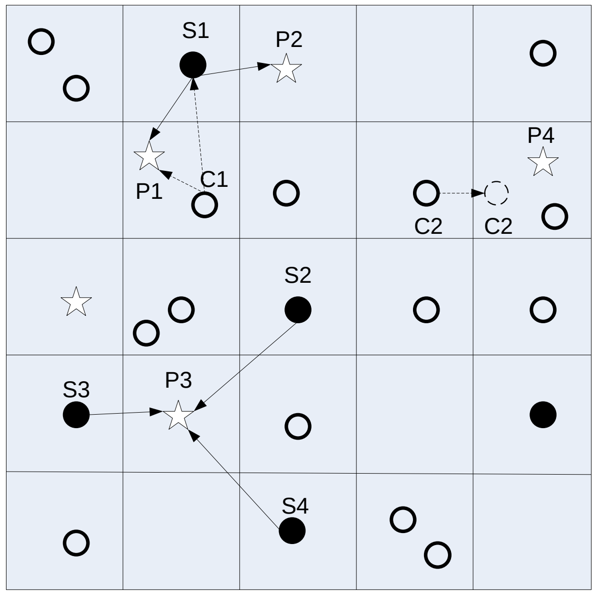


Figure 4.7 An example of moving cost

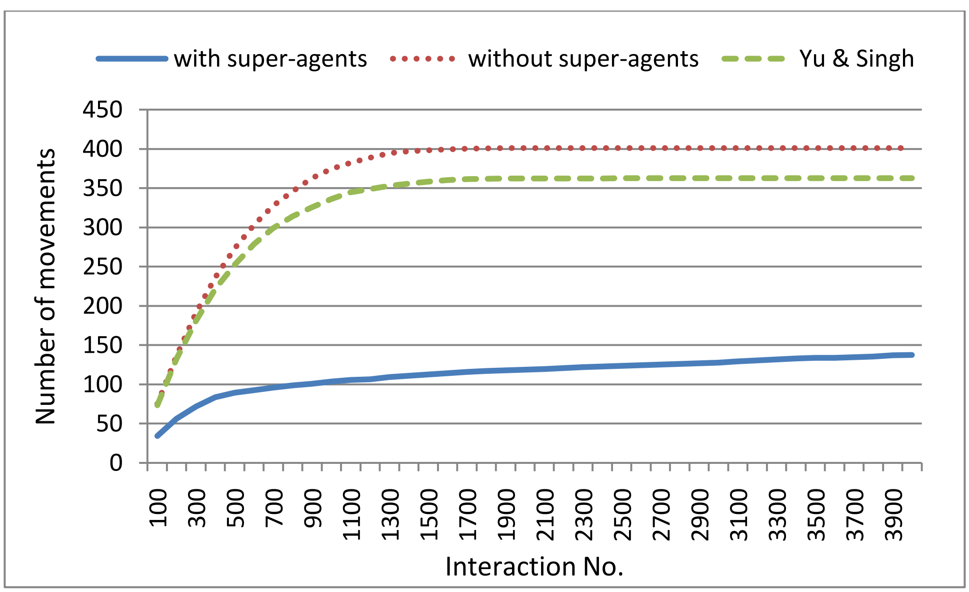


Figure 4.8 Comparison of searching costs

In summary, my super-agent based reputation management shows better effectiveness even when some super-agents are malicious. These results confirm that it is advantageous to make use of the high capabilities of super-agents to maintain reputation of service providers. Their efforts are helpful for other consumer agents in selecting service providers in the environment, in terms of higher probability of finding good providers and saving resources.

4.3.2 Incentives for Service Providers Offering Rewards

In the reward mechanism, rewards offered by service providers serve as motivation for super-agents to build reputation for their services. However, if providers do not provide rewards to super-agents, super-agents have no incentives to build reputation for their services. In the next set of experiments, I demonstrate the great advantages for providers to offer rewards to super-agents and provide incentives to build reputation for their services, which is the important foundation for my reward mechanism to work. For each successful interaction, a service provider will reward the super-agents who provided reputation information to the consumer with one virtual point, which will be equally distributed among them.

In this experiment, I measure the number of consumer agents that trust a service provider. Given a trust threshold, if a consumer agent's trust value in a service provider is greater than the threshold, the service provider is trusted by the consumer agent. I simulate two systems. In one system, all service providers offer rewards to super-agents. In the other system, service providers do not offer rewards and therefore super-agents do not build reputation for their services. Results in Figure 4.9 show that the superior and normal providers that offer rewards to super-agents ("*sup P + reward*" and "*nor P + reward*" in the figure) are trusted by more consumer agents than those that do not offer rewards ("sup P" and "nor P" in the figure) respectively. Therefore, it is beneficial for the superior and normal provides to provide rewards. For the bad providers, rewarding or not rewarding super-agents will not make a difference (see the curves for "*bad P + reward*" and "*bad P*" in the figure).

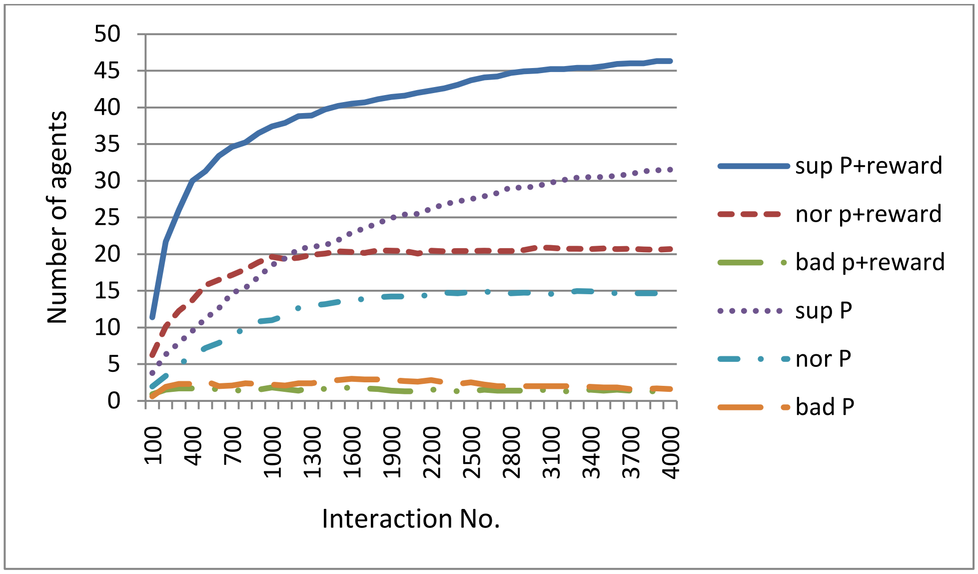


Figure 4.9 Number of agents trusting providers offering reward versus providers not offering reward

I also examine whether offering rewards to super-agents is beneficial for the superior service provider in different situations. The benefit is measured by the number of times that its service is consumed by consumer agents, i.e. the number of interactions with consumers. I simulate the following scenarios:

- All offer reward: all service providers in the system offer rewards to super-agents.
- Only superior providers do not offer reward: the superior service provider does not offer rewards, while all the other service providers offer rewards to super-agents.
- Only superior provider offers reward: the superior service provider offers rewards, while all the other service providers do not.
- All do not offer reward: all the service providers do not offer rewards.

Results are shown in Figure 4.10. By comparing the curves of “all offer reward” and “only superior provider offers reward” with the ones of “only superior provider does not offer reward” and “all not offer reward”, I can see that regardless of whether other service providers will provide rewards, the superior provider is better off by offering rewards. This result provides strong support for my reward mechanism where it is a dominating strategy for good service providers to provide incentives to reward super-agents.

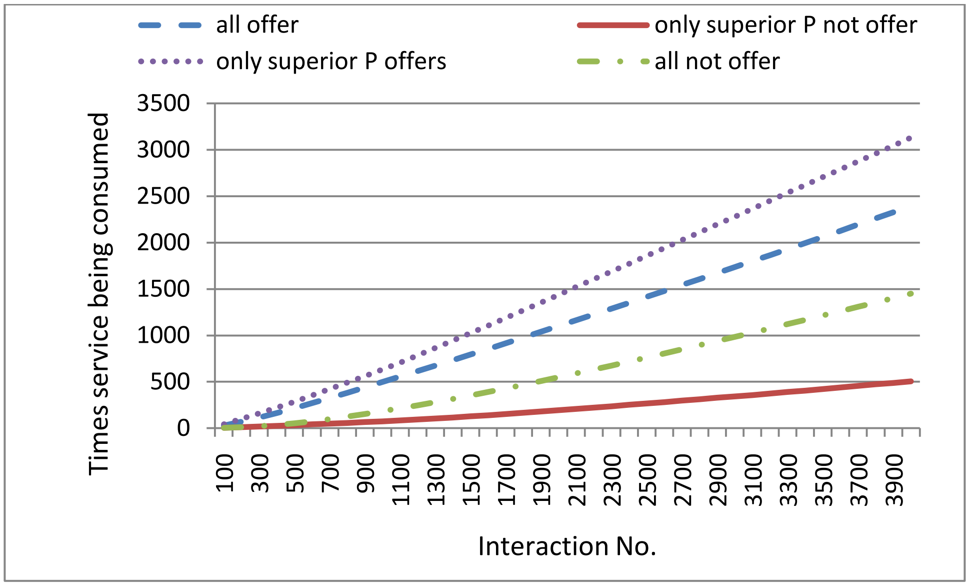


Figure 4.10 Comparison of different situations for superior service providers

4.3.3 Incentives for Super-Agents Being Honest

An important purpose of my reward mechanism is to create incentives for super-agents to provide truthful reputation information about services. In this experiment, I involve some super-agents that are dishonest. I first measure the number of consumer agents trusting super-agents that are honest and dishonest respectively. As shown in Figure 4.11, honest super-agents attract the trust of a large number of consumer agents. Their reputation advice about services will be followed by many more consumers. I also measure the total number of virtual credits that a super-agent can gain when it acts honestly and dishonestly respectively. As shown in Figure 4.12, honest super-agents can gain many more virtual credits than dishonest super-agents. The two lines in Figure 4.12 start to level off at the end of 4000 interactions. This is because with the increase of the number of interactions, agents have gained enough experience to model the trustworthiness of services. They do not need to ask super-agents for reputation information. Therefore, super-agents will receive less reward in this case. Overall, the experimental results support that my reward mechanism provides strong incentives for super-agents to honestly share their reputation information about services with other consumer agents in the system.

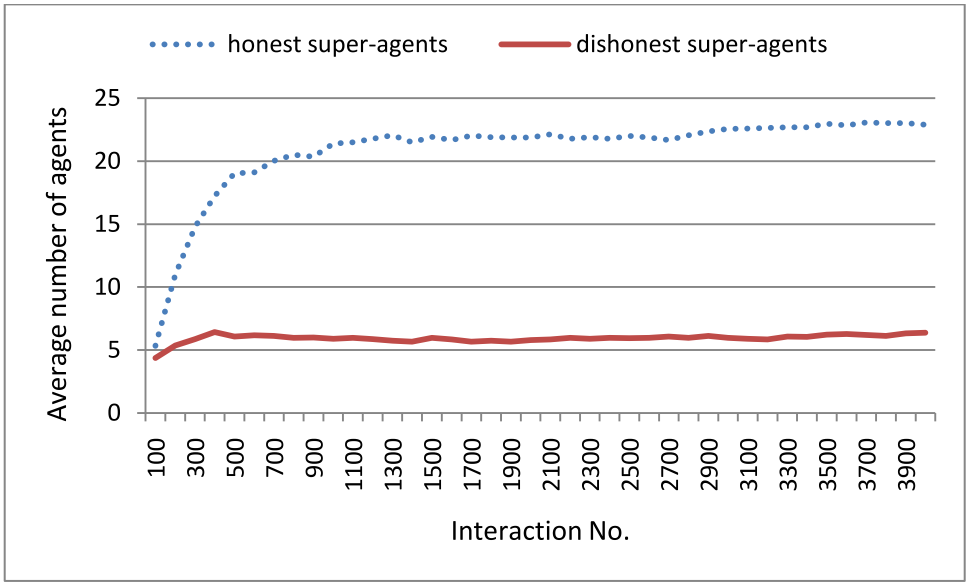


Figure 4.11 Attracting consumer agents: honest super-agent versus dishonest super-agents

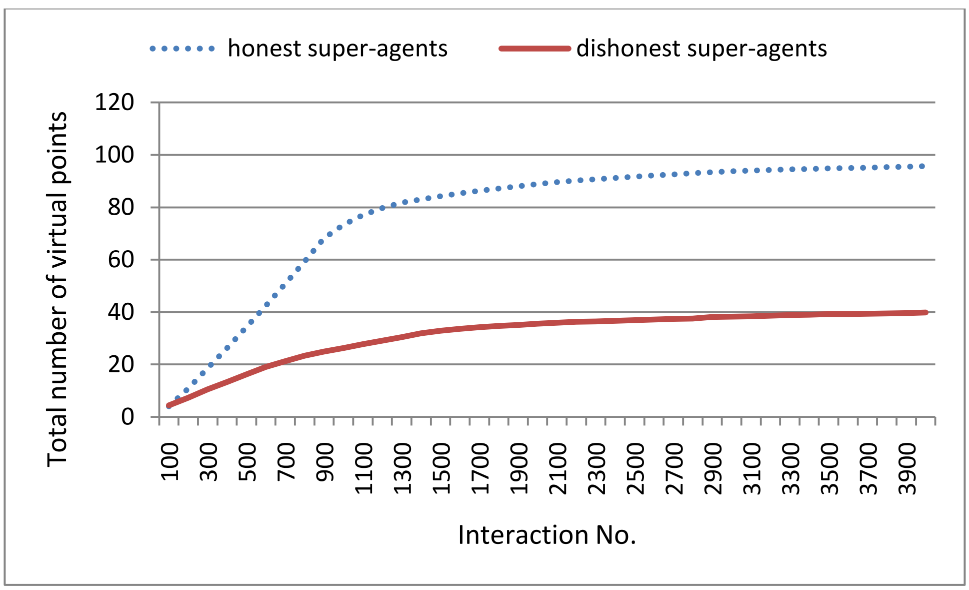


Figure 4.12 Honest super-agents versus dishonest super-agents

4.4 Discussion

My method provides consumer agents with global reputation information about services. Other approaches have also been proposed to get global reputation in a decentralized system, such as EigenTrust [28] and PeerTrust [84]. In EigenTrust, every peer has a reputation manager determined by a hash function. The reputation manager is responsible for computing the global reputation for a peer based on the algorithm of power iteration. This method is very complex and may suffer from scalability problems. PeerTrust introduces several different parameters and factors that can be used to calculate the reputation of peers, and a method of how to combine them. These methods are all designed for a structured P2P network based on, for example, DHT (Distributed Hash Table). A structured network has its shortcomings. It is hard to cope with the situation where agents arbitrarily join and leave. These methods also assume that peers are equal and take the same responsibilities to store and provide information. My approach makes use of super-agents with more capabilities.

My work bears some similarities with the SuperTrust method of Karame et al. [29] and the DIRECT framework of Zhang et al. [92]. In SuperTrust, super-agents play an important role in collecting and storing reputation information for other agents. An agent combines different sources of information (personal experience, friend's opinions and global reputation) to compute the trustworthiness of another agent. Their work emphasizes the importance of using hard security methods (i.e. cryptography) to secure the safety of messages and make sure that the messages are not opened or modified during passing. In DIRECT, a super-agent collects reputation information from other agents and distinguishes honest feedback from dishonest feedback. However, in these methods, an agent's reputation managers (super-agent) are designated by a central "certification authority", and are assumed to be trustworthy. Agents do not have the freedom of selecting their preferred super-agents. In my super-agent based reputation management, the trustworthiness of super-agents is also modeled by other agents in order to select trustworthy super-agents for consultation. A reward mechanism is also proposed to create incentives for super-agents to provide truthful information.

Zhang and Cohen [89] have also proposed a trust-based incentive mechanism for encouraging buying agents to truthfully share information about selling agents with other buying agents in electronic marketplaces. Honest buying agents will become neighbors of many other buyers and be rewarded more by selling agents. Their mechanism relies on a trusted central server to compute the reputation of buying agents based on a social network of buyers and to share this information with selling agents, thus scalability becomes a big concern. However, the super-agent based reputation management and the practical reward mechanism fit naturally in decentralized networks and make good use of capabilities of super-agents distributed across the networks.

4.5 Summary

In this chapter, I proposed a mechanism of using super-agents to manage reputation of services. Super-agents can build a general public reputation opinion about services and share the reputation information with other consumer agents. My approach achieves better effectiveness and scalability in helping consumers find good services. A practical reward mechanism is also designed to encourage super-agents to contribute their resources and truthfully share their reputation information. Super-agents can gain rewards from service providers for their contributions and honest behavior.

Reputation of services built by super-agents reflects a majority consumer agents' opinion. However, agents are different in their interests and judging criteria. The majority opinion may not fit every agent's needs. For example, if an agent is in a minority, reputation information provided by super-agents may mislead the consumer agent to make wrong decisions. In addition, some agents may be picky and the majority's opinion may be not close enough to its own requirement. It would be more desirable to have opinions from agents that are similar in interests and judging criteria. The next chapter explores the idea of forming communities by super-agents that provide community members with more personalized information about services.

Chapter 5

Forming Communities

In Chapter 4, a framework of using super-agents to manage reputation has been proposed. This framework can facilitate agents to find reputation information in a decentralized environment, since super-agents take responsibility in collecting, storing and sharing information, thus acting as reputation servers. The reputation built by super-agents is based on the feedback from all the consumers and reflects the majority agents' opinion. It is called the general public-opinion reputation of the service. However, agents are different in their interests and judging criteria. The majority opinion may not fit all agents' needs. For example, if an agent has some special interests or quality criteria and is thus in the minority, the reputation provided by super-agents may mislead the agent to make decisions that may not be suited to its interests. Such an agent will benefit more from the opinions of agents that are similar to it in interests and judging criteria. This chapter describes an enhancement to the framework described in Chapter 4, where a super-agent can form communities based on its interests and judging criteria. The super-agent will examine the agents who send it feedback and select the agents with similar interests and judging criteria to its own to join its community. In the community, the super-agent will collect evaluations about services from its community members, develop a community-based reputation for services based on the collective opinion of its community members, and share the reputation with the community members. An agent can decide whether to join a super-agent's community. This would happen if the agent considers the community trustworthy, i.e. if its evaluations are similar to the community's evaluations about the services that they both have rated, which reflects similar tastes and preferences between the community and the agent. When an agent has joined the community, it selects services based on the service's reputation information received from the community. If an agent has not joined any community it can use the general public opinion-based reputation to judge the trustworthiness of services. In contrast with the general public opinion-based reputation of services, the community-based reputation of services is based on the opinions from agents with similar interests and judging criteria, not from all of the agents. It will reflect the opinions of agents having similar interests and judging criteria and help these agents make a better selection [69].

In order for the community-based approach to work effectively, super-agents have to contribute resources to maintain communities, build reputation information and answer queries about reputation of services. These super-agents may be malicious in providing reputation information. They may provide false good reputation for some services to promote them or provide false bad reputation to bad-mouth some other services. The reward mechanism applied in Chapter 4 is also used for encouraging super-agents to build communities and behave honestly. Service providers offer rewards to super-agents that bring consumers to consume their services. Super-agents that are honest and contribute more resources will attract a larger number of consumers to join their communities and follow their advice about services. These super-agents will then be able to obtain more rewards from the service providers.

A service selection environment is simulated where some services are of low quality and some agents may be malicious. The experimental results confirm that forming communities results in more effective service selection. They also show the incentives for super-agents to contribute more resources and share truthful reputation information about services and for consumers to honestly provide reputation opinions. The results demonstrate that this approach outperforms the experience-based approach [59] and the model of Yu and Singh [87] when consumers do not have much experience with services.

The rest of this chapter is organized as follows. Section 5.1 presents the definition of community and the motivation and related work. Section 5.2 gives details about the community formation mechanism. Section 5.3 presents the experiment results. Section 5.4 gives some discussion. The last section presents a summary.

5.1 Background, Motivation and Related Work

5.1.1 Definition of Community

Communities exist in human societies, where people with common interests or purposes will get together, share their resources and benefit from them. The word “communities” often appears in the multi-agent systems literature, where agents are used to represent real human users. These are the “virtual communities” composed of groups of human users / agents connected with each other through the Internet. In the multi-agent literature, the term “community” has been used to denote different groups of agents. According to the structural complexity, a community can be classified into three levels.

At the lowest level, the term "community" has been used to refer to a group of agents that cooperate with each other in the same environment, such as "e-commerce communities" [84], "electronic communities" [86], "peer to peer communities" [63]. At this level, the "community" is used as a shorter word to denote the entire multi-agent system.

At the medium level, the term "community" has been used to refer to a (sub-)group of agents that tend to communicate or interact with each other more often than with the remaining agents in the system. In such communities, agents have some kind of proximity and can reach each other within a few hops so that they can easily cooperate. The proximity that defines the community can be identified according to different criteria, such as the link topology [19][20], the interests of agents [32], and the neighborhood of agents [85]. A medium level community can be formed based on pre-defined criteria or by agents themselves through their interaction histories. For example, a community can be formed based on a pre-defined ontology about interests [41]. When an agent joins the system, it can be automatically designated into a community by matching its declared interests with the pre-defined ontology. This is a centralized approach to community formation, assuming that there is a central point in the system that has a list of all available communities and their interest profiles.

Alternatively, communities can form automatically in a decentralized way during the process of agents' interactions without any centralized representation in a self-organizing manner. Agents can gradually link with the agents that they tend to interact more often with and get detached from the agents that they tend not to cooperate with. If agents interact more often with other agents having similar interests, gradually, interest-based communities will evolve. The agents in this kind of community have a self-centered view and do not keep a representation of the community. Neither do they work as a whole to achieve some common goals that will benefit all the members. In fact, the community exists "in the eye of the observer", but is not represented in an explicit way anywhere in the system.

For example, in Yu and Singh's model [1], two kinds of trust are modeled respectively for each agent, expertise and sociability. An agent's expertise refers to the agent's ability to provide required services. An agent's sociability is the agent's ability of suggesting other agents that can provide the required service. Implicit communities are formed, where each agent keeps a list of neighbors from which it can gain good services or referrals. However, it may take a long time for agents to learn about each other and form effective communities.

At the high level, the term "community" has been used in the multi-agent systems literature to denote an explicitly existing organization that facilitates a group of agents with a common goal, interests and preferences to get together, share their knowledge, learn and benefit from one another. In the community, there is an agent responsible for organizing its community members and storing community-related information, called community manager. Community members do not have to be linked close to each other. An agent community formed in this way bears some similarities to some other agent groups that have been studied extensively by the multi-agent research community, like teams and coalitions.

- *Teams of agents* are composed of agents cooperating to solve a particular problem that cannot be solved by any individual member or to solve such a problem more efficiently than it can be solved by individuals on their own. Examples of agent teams are robot-soccer and robot-rescue teams. The main research focus in this area is the assignment of tasks and the coordination among the agents [35][40][44].
- *Coalitions of agents* are mostly used in e-commerce and composed of self-interested agents. The motivation for agents to join a coalition is that they can get more benefit (e.g. discounts) as members of the coalition, although they can also act alone. Coalitions are typically short-term groups. Research on coalitions focuses on how to form coalitions that maximize the individual and group utility and how to distribute the utility among the members [7][14][33][60][91].

In this thesis, the term "*agent community"* is at the high level and used to denote an organization that facilitates a group of agents sharing common interests and preferences to share knowledge, learn and benefit from one another [71]. It focuses on sharing information. A community of agents has similarities with a team and a coalition. Like a team, a community is organized; some agents can take specific community related roles (e.g. the community manager). It is also similar to a coalition. In a community, the agents work together to achieve some common goals, which can facilitate achieving their individual goals. However, the goals of a community are long-term goals and are hard to express in terms of individual utility, which is unlike the goals of agents that enter a coalition. For example, in a P2P network, a community can serve as an information center to provide agents with information that would otherwise be distributed across the different agents. In contrast with teams and coalitions where agents work as a group to interact with other groups, the agents that are members of a community are free to interact with non-members. This is beneficial for the community (allows access to resources outside of the community).

5.1.2 The Benefits of Forming Communities

Forming communities can bring the following benefits.

1. Valuable information

In human society, people often have different opinions about the same thing, say a restaurant. Some people may think a restaurant is good because the food is delicious. Others may think it is bad since a customer has to wait a long time to get service. Different judging criteria may lead to different conclusions. Similarly, when a community is composed of like-minded agents with similar interests and judging criteria, the opinions from the community are more valuable than the general public's opinion, which is an average of the opinions of a large number of diverse agents. This is especially true when an agent belongs to a minority group, since the general public's opinion usually reflects the majority opinion.

2. More information

A community can serve as an information space for agents looking for information. In a centralized system, the central node can list all the resources in the system so that users can browse them and make a selection. However, in a distributed P2P system, there is no such list. Agents have to use keywords to search for resources, which is possible if they have a clear goal and know what specific keywords they can use to find resources. However, most often, agents do not know what keywords to use to find what they need, which could be caused by the lack of related knowledge or having a vague goal [24]. For example, an agent wants to find a good movie to watch. It does not know which movie it should look for. Instead of looking for a specific movie, it searches for a movie community. In the community, it can browse all the movies mentioned by the community members, access their ratings, and finally choose one to watch.

3. Robustness

In human society, people can search for resources through their social networks. A social network is a social structure composed of individuals called "nodes", which are tied by various relationships, such as friendship, common interests, and relationships of trust [80]. These relationships are usually formed through past interactions of nodes. Similarly, agents can also form their social networks through their past experiences and use them to search for resources. Social networks tend to be random [5]. Random networks can be vulnerable to failures. A community, with a shared index of members, creates a form of local centralization, which is similar to that found in scale-free networks, with super-nodes. These types of networks are robust to random failures [5]. To clarify this, Figure 5.1 (a) shows an example – the social network of agent 1, where the arcs denote who knows whom, i.e. if two agents are not connected by an arc, they do not know each other. When agent 1 sends a search request, it will first go to agents 2 and 3. Then they will pass it to agent 4, which will forward it to agent 5 and, finally, to agent 6. If all the agents are available, agent 1's search request can reach all the agents (2, 3, 4, 5, 6). However, if one of the agents, for example, agent 4 is offline or not available at the time of request, agent 1 has no connection with agent 5 and 6, even though they are available, so the request will not reach them. This situation will not happen in a community with an explicit index of members. When all the agents are in the same community as shown in Figure 5.1(b), even if one of them is unavailable, the rest of them can still know and reach each other through the index kept by the super-agent that maintains the community. Of course, this assumes that the super-agent is available all the time. Scale free networks in general are not sensitive to random failures, but they are very vulnerable to targeted attacks to the super nodes [5].

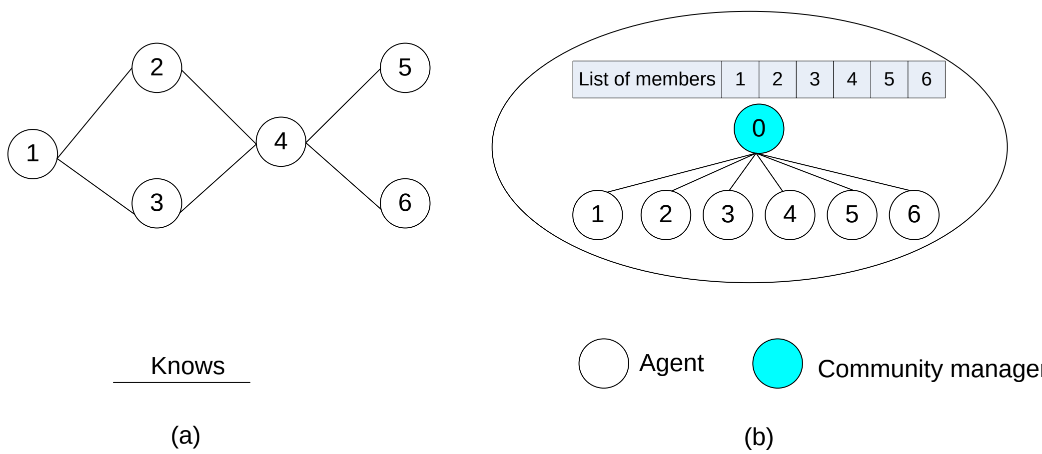


Figure 5.1 A social network vs. a community

4. The benefit for service providers

When agents act as service consumers, they can benefit from forming communities by getting more valuable information. When they act as service providers, they can also benefit from the community. A service provider can analyze the features of a community and fine tune its offerings for the particular community [85].

5.1.3 Related Work

A strong motivation for my work is that forming communities can help agents to find more valuable information. Agents often have different opinions about the same thing because of subjective differences (different judging criteria). A community is composed of like-minded agents with similar interests and judging criteria. The opinions from the community members are more valuable than the general public's opinion.

Different approaches have been proposed for coping with subjective differences among consumer agents. For example, Regan et al. [49] propose a Bayesian modeling approach to allow a consumer to learn other consumers' evaluation functions on different features of the services delivered by providers. This is done by analyzing ratings that are provided by the other consumers for the services. The authors claim that this makes it possible to adjust provided ratings for any subjective differences. Sensoy et al. [59] develop an approach for distributed service selection that allows consumers to represent their experiences with the service providers using ontologies. An experience is a record of what service the customer has requested and received in return. In this way, the experience-based approach allows the objective facts of the experiences (other than subjective opinions, i.e. ratings) to be communicated to the other party and thus eliminates subjective differences among consumers. However, these two approaches require consumer agents to either learn complicated models of other consumers or represent their experiences using ontologies. My proposed community-based approach does not require extra effort from consumers. In Section 5.3.3, I will demonstrate the benefit of forming communities and compare with the experience-based approach of Sensoy et al. [59] through experiments.

As mentioned in Section 5.1.1, communities at the medium level can be either formed explicitly based on pre-defined ontologies in a non self-organizing way [41] or formed implicitly in a self-organizing way by agents themselves through their interaction histories (e.g. Yu and Singh's method). In contrast to these two approaches, I allow super-agents to create and maintain explicit communities in a self-organizing way. For the approach using pre-defined ontologies, the non self-organizing method, it cannot deal with the problem of subjective differences of agents, since agents may have the same interests, but different judging criteria. But for the implicit community formation approach of Yu and Singh [86], it requires agents to model each other and communities will be formed slowly. My super-agent based approach can not only deal with the problem of subjective differences, but also offloads duties from the individual consumers and allows forming communities faster, which is demonstrated in Section 5.3.3 through an experiment comparing it with Yu and Singh [86]. A similar framework, called Surework, is proposed in [51] where super-peers form clusters of ordinary peers. However, the authors did not provide any computation details nor did they explore the benefit of providing more valuable information for peers.

5.2 A Super-Agent Based Community-Building Mechanism

5.2.1 The Overview

A super-agent can form its own community around its interests and judging criteria. For example, in a web service system, a super-agent can build a community for travelers sharing information about related services, such as car rental services and air ticket booking. In a P2P file sharing system, communities can be built for music fans to share music files in a particular genre or movies featuring a particular actor. In order to build a community, a super-agent has to meet some minimal requirement about its resources, such as CPU power, bandwidth, storage room and availability. The number of members of the community depends on the amount of resources a super-agent has dedicated to it.

A super-agent, called *community manager*, is responsible for selecting community members. Since a community manager has limited resources, it can only form a community containing a limited number of members. Therefore, a community manager has to be selective and choose as members only the agents that it regards the most trustworthy, since they are more likely to provide valuable information that can benefit the community manager and members. The community manager collects and stores ratings from its community members about the services they have interacted with, builds community-based reputation values for the services and shares the information with the community members. The detailed formalization of these responsibilities and processes is provided in Section 5.2.4.

A consumer agent may join multiple communities created by multiple super-agents. If these super-agents share similar interests and judging criteria, they build similar communities. Then, they may have some community members in common and have some overlapping information, which brings beneficial redundancy to the system. Since a community manager is the core of its community and responsible for storing all the community-related information and maintaining the community, once the community manager is not available, the information stored in the community will also not be accessible. Redundant information stored in other communities will work as a backup and make the whole system more robust.

In this section, I first provide a description about my community-based service selection approach. The process of searching and selecting a service is similar to that in Chapter 4 (see section 4.1.1.2). The difference is that when a consumer agent wants to find a service, it will search services not only through the system's searching protocols, but also through the managers of the communities the agent belongs to. When a community manager receives such a query, it will check whether its community is building reputation for services matching the search keywords. If yes, they will send to the consumer the services' information (e.g. the names and descriptions of the services). They may also be asked for community-based reputation values of the services. Based on the received reputation values of the services, the consumer can then model the trustworthiness of the services based on the reputation information from the communities in which it participates. If a consumer agent does not belong to any community or its communities do not provide reputation information about a particular service, it will model the trustworthiness of the service using the method described in Chapter 4. The formalization for calculating the trustworthiness of a service based on reputation information from communities is presented in the next section.

5.2.2 Trustworthiness of a Service

When a consumer agent *p* judges the trustworthiness of a service *s*, it will first use its own experience. If *p* does not have enough personal experience with *s*, it will consider community-based reputation information about the service provided by super-agents. The agent sorts the list of communities according to its trust in them from high to low. The modeling of the trustworthiness of communities is described in the next section 5.2.3. If the agent's trust in a community is higher than a threshold, the super-agent that has formed the community is regarded as trustworthy and will be asked for community-based reputation of the service, which is a value in the interval [0,1] where 0 means that the service is totally disreputable and 1 means that the service is completely reputable. Once the consumer receives all community-based reputation values of the service from all trustworthy communities, (c_1, c_2, ..., c_h), the consumer agent will calculate an aggregated reputation value according to the following weighted average formula:

$$R_c(s) = \frac{\sum_{i=1}^{h} T(c_i) * R(c_i, s)}{\sum_{i=1}^{h} T(c_i)} \tag{5.1}$$

where $T(c_i)$ is the consumer agent's trust in the community c_i formalized in the next Section 5.2.3 and $R(c_i, s)$ is the community-based reputation of service s provided by c_i formalized in Section 5.2.4.

The trustworthiness of service s $T(s)$ is calculated based on the combination of the consumer agent's trust $T'(s)$ in the service calculated using its own experience and the aggregated reputation value from the community, $R_c(s)$, as follows:

$$T(s) = w * T'(s) + (1 - w) * R_c(s) \tag{5.2}$$

$T'(s)$ is a representation of an agent's past experiences and shows the extent of trustworthiness that an agent has in the service based on its own experiences. It is modeled using the Bayesian network-based method as described in the section 4.1.2. An agent can get different values for $T'(s)$ from its Bayesian network depending on its requirements. For simplicity, in my simulation, as the trust value that an agent has of a service is computed by the percentage of the satisfying interactions of all interactions with the service.

w represents how much weight should be put on $T'(s)$. When w equals 1, the trustworthiness of the service is the same as the trust value calculated based on only the agent's personal experience with the service. When w is less than 1, the aggregated reputation $R_c(s)$ of the service also plays a role in the calculation of the trustworthiness of the service. The value of w is determined based on the number of interactions between the agent and the service s. The method for determining the value of w is the same as that described in section 4.1.3.

Note that there may be the case where a consumer agent does not join any community or does not know which community is good to join. In this case, it will use the general public opinion-based reputation from super-agents to judge the trustworthiness of services which is described in

5.2.3 Trustworthiness of a Community

The trustworthiness of a community is calculated to determine whether a consumer agent trusts it and would like use its community-based reputation of a service. It is also used to determine how much weight should be put on each community-based reputation value in Equation 5.1. If the communities the consumer belongs to are untrustworthy, the consumer may want to leave the communities. After a consumer agent joins a community, it can get a community based reputation value of a service from a community manager (the super-agent who formed the community), and it can develop trust in the community based on its experience of using the service. After each time of using a service, the consumer can evaluate its experience *e(s)* as "satisfying" or "not satisfying" (1 or 0 respectively). The reinforcement learning formula is used to model the trustworthiness of the community, as follows:

$$T(c_i) = \alpha * T'(c_i) + (1-\alpha) * e(c_i) \tag{5.3}$$

where c_i denotes the *i-th* community. $T(c_i)$ denotes the consumer agent's trust in the community c_i after the update, and $T'(c_i)$ denotes the trust value before the update. $e(c_i)$ is the evaluation of the consumer agent's current experience with the advice provided by the community c_i about the service.

$$e(c_i) = \begin{cases} R(c_i\,, s)\,, & if \;\; e(s) = 1 \\ 1 - R(c_i\,, s), & if \; e(s) = 0 \end{cases} \tag{5.4}$$

To explain, the value of $e(c_i)$ is determined by comparing the consumer agent's own experience of using the service, *e(s)*, with the community-based reputation about the service provided by c_i. If the consumer agent's experience of using the service is satisfying ($e(s) = 1$), $e(c_i)$ is equal to the reputation value provided by the community manager about the service, which is $R(c_i, s)$. If the consumer agent's experience of using a service is not satisfying ($e(s) = 0$), $e(c_i)$ equals $1 - R(c_i, s)$. A community can gain more trust if the community based reputation value matches more closely the consumer agent's experience. The initial value of a consumer agent's trust in a community may be set to 0.5, which means that the community is neither trustworthy nor untrustworthy.

5.2.4 A Super-Agent Based Community Formation

Since a community manager (super-agent) has limited resources, its community can only contain a limited number of members. Therefore, the manager has to be selective and choose only the agents that it regards the most trustworthy as members. These agents are more likely to provide valuable information that can benefit the manager and other community members.

1) Selecting community members

In Chapter 4, a super-agent can serve as a reputation manager for a service that it is interested to collect feedback from consumer agents and develop a general public-opinion based reputation for the service. The super-agent can also form its community and select agents to join its communities from the agents who send it feedback. At the beginning of the process of forming a community, the super-agent can ask a consumer agent *p* to send it feedback (its trust value) for each encountered service. Then the super-agent can evaluate the agent to decide whether to add it into the community. If the evaluation value exceeds a predefined threshold, the agent will be selected as a member. An invitation will be sent to it.

The evaluation of an agent, called the agent's reputation, is a measure of how much the agent is trusted by the super-agent and its community members. So the consumer agent's reputation in the community *R(sp,p)* is determined by two components, the community manager's trust *T(sp , p)* and the average trust of community members in the consumer, as follows:

$$R(sp,p) = w'T(sp,p) + (1-w')\frac{\sum_{i=1}^{n} T(cm_i,p)}{n} \quad (5.5)$$

$$w' = \begin{cases} \frac{k}{N'_{min}}, & if\ k \le N'_{min} \\ 1, & otherwise \end{cases} \quad (5.6)$$

cm_i denotes the *i-th* community member. $T(cm_i, p)$ denotes a community member cm_i's trust in *p*, and *n* is the total number of community members. The community manager also assigns different weights (*w*', a value between 0 and 1) to the two components. The community manager *sp* models the trustworthiness of the consumer agent *p* based on their ratings for their commonly rated services. If the super-agent has more common rated services with the agent *p,* it can be more confident about *p's* similarity with it in terms of interests and judging criteria. It will put more weight on its own judgment. The value *w'* can be determined by formula 5.6. *k* is the number of common rated services by both sp and p. N'_{min} is the minimum number of common rated services required for sp to be fully confident about p's similarity.

$T(sp, p)$ represents the similarity between sp and p, which can be measured based on the average absolute deviation between their opinions about common rated service. As shown in Formula 5.7. s_j denotes the *j-th* common rated service. $T_{sp}(s_j)$ is sp's trust in the service s_j. $T_p(s_j)$ denotes p's trust in s_j. $T(cm_i, p)$ represents the similarity between the community member cm_i and p. The way of calculating $T(cm_i, p)$ is similar to that of $T(cm_i, p)$.

$$T(sp, p) = 1 - \frac{\sum_{j=1}^{k} |T_{sp}(s_j) - T_p(s_j)|}{k} \quad (5.7)$$

The agent who has been asked to join the community also has to decide whether it wants to do it. The super-agent will provide the agent the reputation of services built by its community, so that the agent can use the same way to compare its evaluation with the community's evaluation (i.e. the community-based reputation) about their common rated services to decide whether to join the community.

2) Updating community membership

A super-agent will update the list of the community members periodically. This is necessary, because a community member may be reputable before joining the community but may become less reputable afterwards, due change of interests or judging criteria. The super-agent sorts all the agents by their reputation values. The number of agents in the community that can be supported by the manager defines the reputation threshold for membership in the community. If a community member's reputation falls below the threshold, a request to leave the community will be sent to the agent. No further updates from this agent will be considered by the community manager and members. The agent will not get information from the community any more.

3) The community-based reputation

The community-based reputation for a service is the average trust in the service from all the community members (including the community manager) as formula 5.8 shows. $R(sp, s)$ is the community-based reputation for the service s. $T_{sp}(s)$ is the community manager's trust in s. T_{cm_i} is the trust in s from the *i-th* community member. n is the total number of community members (not including the community manager). If a member happens to have no interaction with a service, a default trust value is used to represent its trust for the service in the formula. The default trust value for a service is the same for all the consumers and services. A super consumer remembers all the trust values from all the members. When a member updates its trust in a service, it will automatically send its updated trust value to the super-consumer. Then the super-consumer will store the value and use it to update the community-based reputation for the service. In this way, a super consumer collects updated information from its members when the members are available and make their updates.

$$R(sp, s) = \frac{T_{sp}(s) + \sum_{i=1}^{n} T_{cm_i}(s)}{n+1} \tag{5.8}$$

5.2.5 The Reward Mechanism

In the system, super-agents have to contribute more resources to maintain communities, model community-based reputation of services, and answer queries of consumer agents. They need incentives for contributing resources. In addition, some super-agents may be dishonest in providing reputation information. They may provide false good reputation for some services to promote these services or provide false bad reputation to bad-mouth some other services. To address these two problems, the reward mechanism used in Chapter 4 is also applied to encourage super-agents to form communities and share truthful reputation information about services. In the reward mechanism, service providers provide rewards to community managers. Each provider issues its own “virtual points”. If a consumer agent is satisfied with a service after consuming it, the consumer agent will send the provider the list of super-agents (community managers) from whom the consumer agent chooses to ask for information about the service’s reputation. A number of “virtual points” will be awarded by the service provider to these super-agents. However, if the consumer agent is not satisfied with the service, it will not provide the list of super-agents to the service’s provider. Therefore, no super-agents will be rewarded. This will prevent super-agents from gaining rewards by providing fake good reputation of services. The number of “virtual points” may be dependent on the value of the service consumed and on the total number of super-agents reported by the consumer agent. The “virtual points” are equally distributed among the trustworthy super-agents.

These “virtual points” issued by a service provider can be used to redeem services offered by this provider. The “virtual points” may also be used to give super-agents higher priority to consume services or provide them with higher quality of services. Therefore they provide an incentive for the super-agents to create communities and contribute their resources to compute reputation of services.

Super-agents have incentives to form communities computing reputation for both good and bad services. For community managers, if their communities build reputation for good services, they can gain “virtual points” from the providers of these good services. The super-agents can then redeem the points for their future interactions with the service providers, i.e. consuming the good services. If some services are bad, super-agents may not gain “virtual points” from the providers of these services because consumer agents will likely not consume these services. However, it is still beneficial for super-agents to build reputation for bad services. They can gain trust from consumer agents by reporting honestly the bad service’s reputation. This can potentially increase the super-agents’ chance of being asked for advice by the consumer agents and the ability to gain points from good service’s providers (in case the super-agents also build reputation for these good services). If a super-agent contributes more resources to maintain communities, build reputation information about services, and truthfully share the reputation information with consumer agents, it will be trusted by many consumer agents and have a larger number of community members. It is then able to bring more consumer agents to consume good services. Their good behavior will be rewarded by the service providers offering good services with virtual credits that the super-agent can use to consume the good services for which it builds reputation.

On the other side, service providers in the system also have an obvious incentive to provide rewards to super-agents. The communities will help the service providers propagate their service information and therefore will bring more consumers.

5.3 Experiments

In this section, I carry out several experiments to evaluate my community-based service selection approach. I demonstrate the benefit of forming community for more effective service selection. I also show the incentives created by my system for super-agents to contribute more resources in forming communities and building community-based reputation for services, and for consumer-agents to be honest. I finally compare my community-based approach with the experience based approach [59] and the model of Yu and Singh [86].

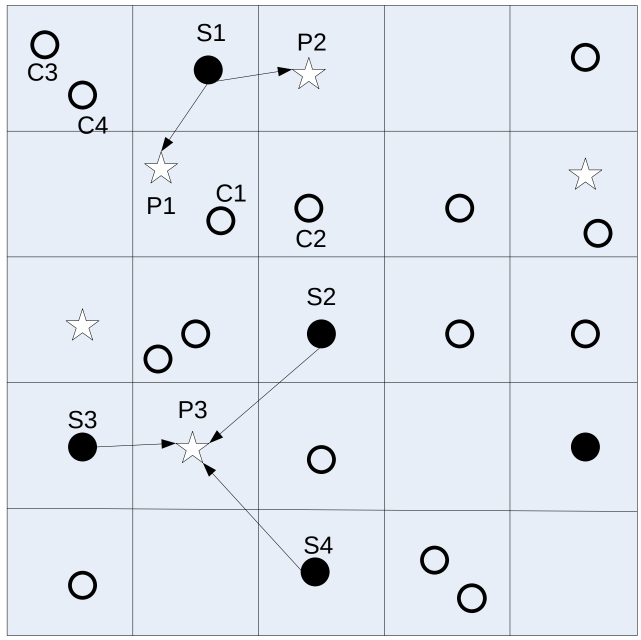


Figure 5.2. A simulated service selection environment

A service selection environment is simulated, similar to that in Section 4.3. It involves service providers and consumers, some of which are super-agents. Consumer agents and super-agents both consume services provided by service providers. A matrix with 5×5 cells is used to simulate a peer to peer (P2P) system as shown in Figure 5.2. The accessibility of peers in P2P environments is mapped to the matrix. Agents in the same cell are neighboring peers that can reach and communicate with each other by one or more hops. Originally, service providers (shown as stars), consumer agents (shown as white circles) and super-agents (shown as black circles) are randomly located in the cells. Consumer agents and super-agents are different in their ability in discovering service providers. Consumer agents can find directly the service providers in their own cell. In this way, the limited connectivity of consumer agents is simulated. A consumer agent will only connect with a few other consumer agents in the same cell. Super-agents will connect with more agents. Super-agents are able to directly find the service provides not only in their own cells, but also in the cells adjacent to their own cells. For example, in Figure 5.2, consumer agent *C1* can only find directly provider *P1* but not *P2*. Super-agent *S1* can directly find both *P1* and *P2*. This simulates that super-agents have more searching power than ordinary consumer agents in the network. Super-agents can build general public opinion based reputation for services provided by the service providers within their searching scope, which is the same as the simulation in Chapter 4. For example, super-agents *S1* will build reputation for both services, *P1* and *P2*. Super-agents *S2*, *S3* and *S4* all build reputation for a service provided by *P3*. In addition, here super-agents can also create their own communities and build community-based reputation for services based on their community members' opinions. The following shows an example of how a super-agent can form a community and build community-based reputation of services.

As shown in Figure 5.3, consumer agents, *C1*, *C2*, *C3*, *C4*, can find both services provided by *P1* and *P2* through super-agent *S1*. Let's suppose for simplicity that each service provider only provides one service. Suppose the super-agent *S1* and consumer agents *C1*, *C2*, *C3*, *C4* all use the service *P1* and *P2* and give their evaluations as shown in Table 5-1, where *"1"* means a satisfying interaction and *"0"* represents an unsatisfying interaction.

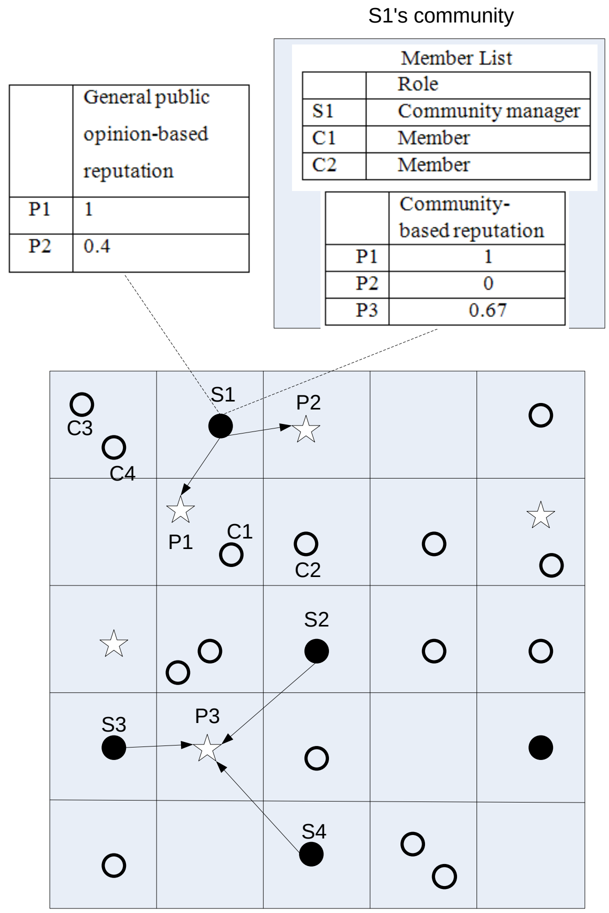


Figure 5.3 An example of forming a community

Table 5-1 Evaluations about the services P1 and P2

	S1	*C1*	*C2*	*C3*	*C4*
P1	*1*	*1*	*1*	*1*	*1*
P2	*0*	*0*	*0*	*1*	*1*

The super-agent *S1* can model the general public opinion-based reputation of *P1* and *P2* based on the evaluations from all the consumer agents, *S1* and *C1*to *C4*.

$$R(S1, P1) = \frac{the\ number\ of\ interactions\ that\ are\ satisfying}{the\ total\ number\ of\ interactions} = \frac{5}{5} = 1$$

$$R(S1, P2) = \frac{the\ number\ of\ interactions\ that\ are\ satisfying}{the\ total\ number\ of\ interactions} = \frac{2}{5} = 0.4$$

R(S1, P1) and *R(S1, P2)* represent the general public opinion-based reputation of *P1* and *P2* from super-agent *S1*.

After using a service, consumer agents can develop their trust in a service. Although they can model their trust using the Bayesian network based method as described in Chapter 3, here instead, for the sake of simplicity, the probability of successful interactions will be used as a measure of the consumer agent's trust in a service, which is shown in Table 5-2. The symbol $T_a(b)$ represents *a*'s trust in *b* (i.e. *P1* or *P2*). *a* could be *S1*, *C1*, *C2*, *C3* or *C4*. For example, $T_{S1}(P1)$ denotes *S1*'s trust in *P1*.

$$T_{s1}(P1) = \frac{the\ number\ of\ successful\ interactions}{the\ number\ of\ interactions} = \frac{1}{1} = 1$$

Table 5-2 Trust in the services P1 and P2

	S1	C1	C2	C3	C4
P1	$T_{S1}(P1) = 1$	$T_{C1}(P1) = 1$	$T_{C2}(P1) = 1$	$T_{C3}(P1) = 1$	$T_{C4}(P1) = 1$
P2	$T_{S1}(P2) = 0$	$T_{C1}(P2) = 0$	$T_{C2}(P2) = 0$	$T_{C3}(P2) = 1$	$T_{C4}(P2) = 1$

According to formula 5.7 (on page 97), super-agent *S1* can calculate its similarity with consumer agent *C1* to *C4* by comparing its trust value with their trust value about the service *P1* and *P2*. *T(S1,C1), T(S1,C2), T(S1,C3)* and *T(S1,C4)* represent the similarity between *S1* and *C1, C2*, *C3* and *C4*.

$$R(C1) = T(S1, C1) = 1 - \frac{|\mathrm{T_{S1}(P1) - T_{c1}(P1)}\,| + |\mathrm{T_{S1}(P2) - T_{c1}(P2)}|}{2} = 1$$

$$R(C2) = T(S1, C2) = 1 - \frac{|\mathrm{T_{S1}(P1) - T_{c2}(P1)}\,| + |\mathrm{T_{S1}(P2) - T_{c2}(P2)}|}{2} = 1$$

$$R(C3) = T(S1, C3) = 1 - \frac{|\mathrm{T_{S1}(P1) - T_{c3}(P1)}\,| + |\mathrm{T_{S1}(P2) - T_{c3}(P2)}|}{2} = \frac{1}{2}$$

$$R(C4) = T(S1, C4) = 1 - \frac{|\mathrm{T_{S1}(P1) - T_{c4}(P1)}\,| + |\mathrm{T_{S1}(P2) - T_{c4}(P2)}|}{2} = \frac{1}{2}$$

Suppose the minimal number of common rated services N'_{min} is 2. According to formula 5.6, the reputation of *C1* and *C2*, *R(C1)* and *R(C2),* are 1 and the reputation of *C3* and *C4*, *R(C3)* and *R(C4),* are 0.5. If the reputation threshold for *S1* selecting a community member is greater than 0.5, *C1* and *C2* will be chosen as members of *S1*'s community. The community based reputation of the service *P1* and *P2*, *R(S1, P1)* and *R(S1, P2),* are 1 and 0 according to formula 5.8.

$$R(S1, P1) = \frac{T_{s1}(P1) + T_{c1}(\,P1) + T_{c2}(\,P1)}{3} = \frac{1 + 1 + 1}{3} = 1$$

$$R(S1, P2) = \frac{T_{s1}(P2) + T_{c1}(\,P2) + T_{c2}(\,P2)}{3} = \frac{0 + 0 + 0}{3} = 0$$

The community allows a super-agent and the members to learn about a wider range of services that may have been experienced by some of the members. For example, the consumer agent *C2* can also find service *P3* through the super-agent *S2*. Suppose *C2* uses the service *P3* and has a successful experience. Since *C2* has joined the community built by *S1*, it will share its new experience about *P3* (i.e. its trust in *P3*) with the community. The calculation of the trust value of *C2* in *P3* is similar to $T_{S1}(P1)$, which is *1* in this case. Since *S1* and *C1* have no experience with *P3,* the default value *0.5* (neither trustworthy nor untrustworthy) is used as their trust in *P3*.

$$R(S1, P3) = \frac{T_{s1}(P3) + T_{c1}(\,P3) + T_{c2}(\,P3)}{3} = \frac{0.5 + 0.5 + 1}{3} = 0.67$$

Suppose in the simulation, there are four types of services provided by service providers, and 2 services for each type. Different types of services have different service qualities varying from very low quality to high quality. There are three types of consumer agents: non-picky, middle-picky and picky consumers. Each type of consumer judges the quality of each type of service differently according to Table 5-3. For example, picky consumers consider as good only services of high quality. For non-picky consumers, almost all the services except the services of very low quality are good.

Table 5-3 The service types and judgements

	Non picky	Moderately picky	Picky
Service Type 1: low quality	Bad	Bad	Bad
Service Type 2: low quality	Good	Bad	Bad
Service Type 3: moderate quality	Good	Good	Bad
Service Type 4: high quality	Good	Good	Good

The simulation involves 100 consumer agents, including 30 picky consumer agents, 40 middle-picky consumers, and 30 non-picky consumers. The number of super-agents accounts for 10% of each type of consumers. Accordingly, there are 3 picky super-agents, 4 middle-picky super-agents and 3 non-picky super-agents. In the initial state of my simulation, consumers have no knowledge of the service qualities. There are 4000 interactions in the simulation. In each interaction, a consumer agent selects and uses a service. I run each experiment for 10 times and present the average of the results.

5.3.1 Demonstrating Benefit of Forming Communities

I first carry out a set of experiments to demonstrate the benefit of forming communities. In the first experiment, I compare the overall performance of two systems. One system simulates the system described in Chapter 4, which does not form communities. In this system, each super-agent collects reputation opinions to build a general public opinion based reputation for services. In the other system, super-agents not only build the general public opinion based reputation, but also form communities, build and share the community-based reputation of services. In the system, if a consumer agent joins a community, it can use community-based reputation to select services. Otherwise, it will use the general public reputation for services.

I measure the performance of a system based on the ratio of successful interactions. A successful interaction means that a consumer agent selects a service to use and finds it satisfying. By using this measure, I can find out whether forming communities can actually help consumer agents find satisfactory services to consume. Figure 5.4 shows the ratio of the number of successful interactions over the total number of interactions. This figure shows that the community based system performs better than the system that does not form communities. Forming communities can help consumer agents more accurately find satisfactory services.

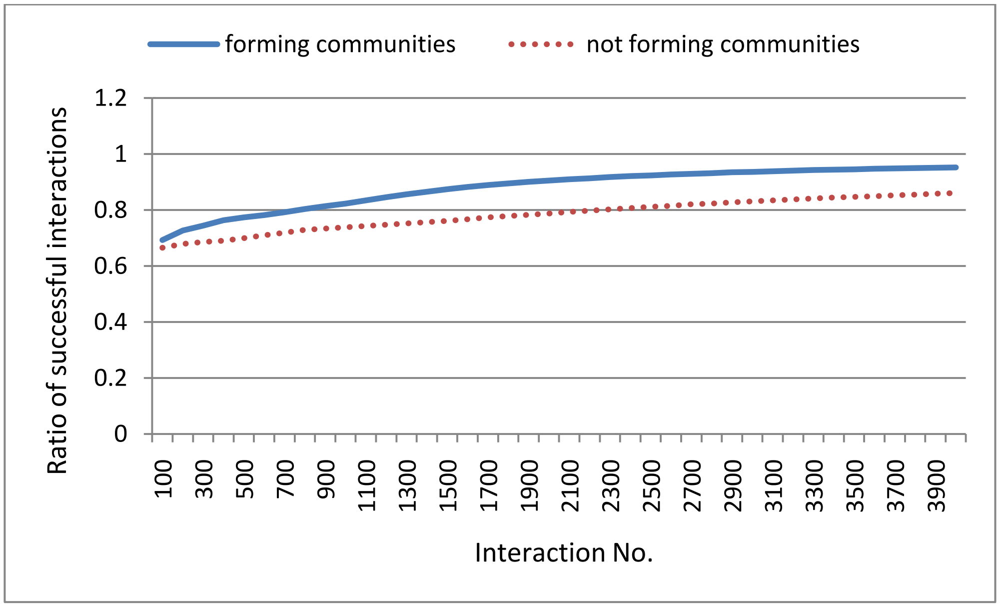


Figure 5.4 Overall performance for systems with and without communities

I further check the performance of each type of consumer agents in the two systems. The results in Figure 5.5 show that picky and middle-picky consumers perform better in the system with communities. The advantage of using a community is particularly obvious for picky agents. They represent the users that have high standard requirements or the users that have some special needs and belong to a minority group. Forming communities benefit them most. However, the group of the non-picky consumers can not benefit much from forming communities. They perform almost the same in the two systems, which is expected because almost every service is good for this type of consumer. They represent the type of user who are easy to please or whose requirements are lower than the average.

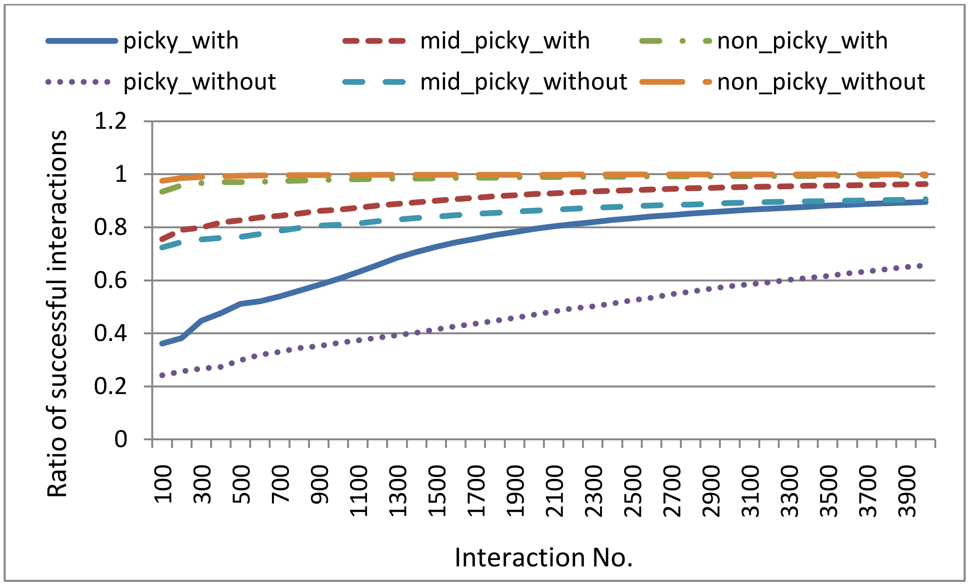


Figure 5.5. Performance of different types of consumers

The second experiment is to show the benefit for consumers to join a community when communities are formed. From each type of consumer agent, a consumer agent is randomly selected which does not join any community. I compare the successful interaction ratio between the selected consumer agent which does not join communities with the rest of consumer agents in the same agent group which join communities. If a consumer agent does not join a community, it cannot acquire community-based reputation information about services from super-agents. The results in Figure 5.6 show that consumers joining communities will gain higher successful interaction ratio. It is thus beneficial for consumers to join communities. The results also show that picky consumers and middle-picky agents benefit the most from joining communities. Non-picky agents also benefit at the beginning by joining communities.

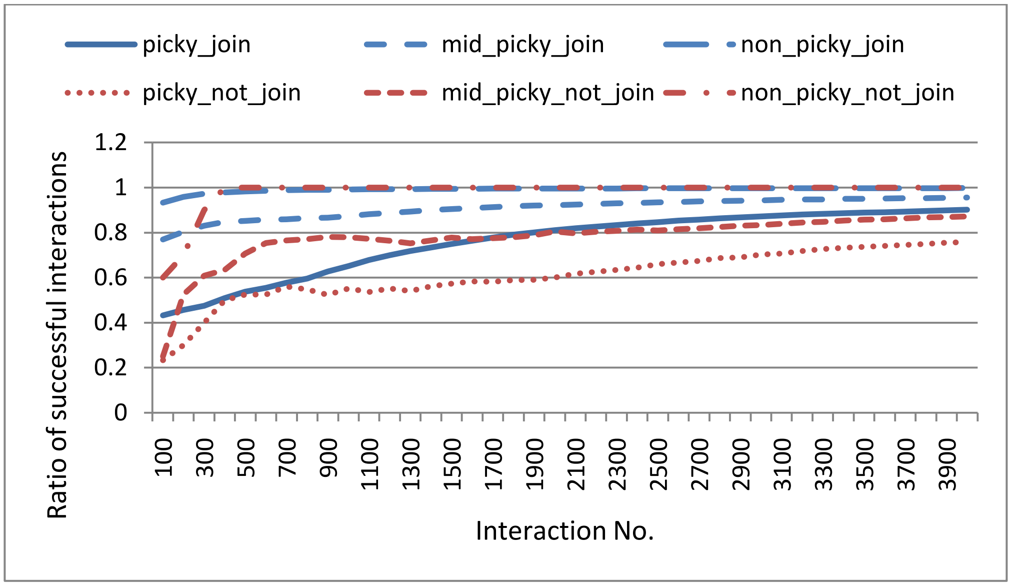


Figure 5.6 Consumers join vs not join communities

5.3.2 Incentives

I also carry out a set of experiments to show the incentives created by my system. In the first experiment, I show the greater gain for super-agents to contribute more resources and build community. I compare the rewards that a super-agent receives when building and not building communities respectively. In the reward mechanism, when a super-agent helps a consumer agent find a satisfactory service to consume, the consumer will report to the service's provider. The provider will then reward the super-agent. If a consumer agent reports multiple super-agents to a service provider for one successful interaction with the service, the reward will be equally distributed among multiple super-agents. For simplicity, for each successful interaction, a service provider will reward one virtual point to all super-agents reported by the consumer. Note that when a super-agent does not build a community, it still provides a general public's reputation value about services to consumer agents, in order to gain some rewards, so it may also help a consumer find a good service and will also be awarded virtual points. The results in Figure 5.7 show that building communities can bring more rewards to super-agents. Therefore, clearly, the reward mechanism creates incentives for super-agents to contribute resources to form communities and build and share community-based reputation of services.

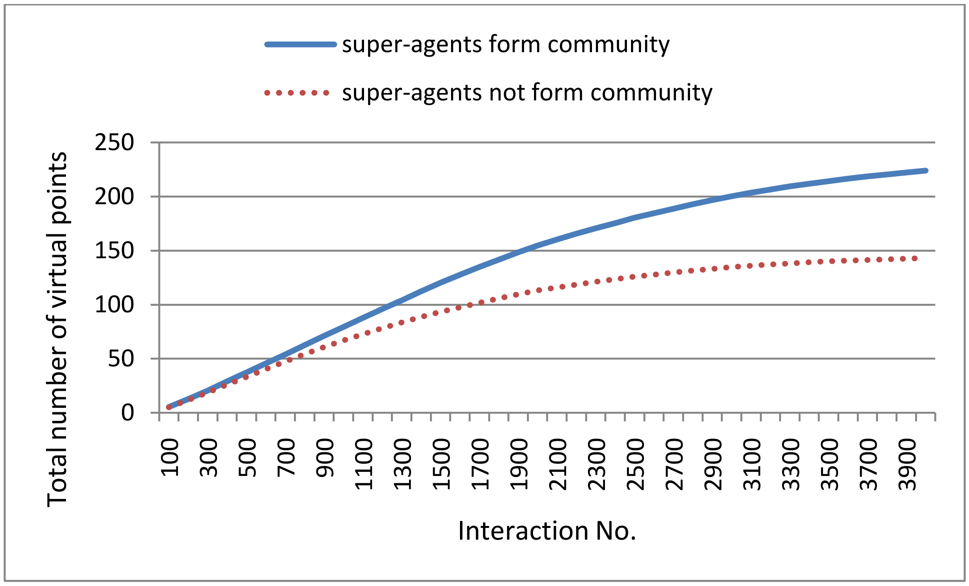


Figure 5.7 Incentives for super-agents forming communities

Another important purpose of the reward mechanism is to create incentives for super-agents to provide truthful community-based reputation information about services. I involve 50% of super-agents that are dishonest and always provide opposite reputation information. For example, if a service's reputation is 0.4, the opposite reputation value will be 0.6 (i.e. 1-0.4). I measure the average number of virtual credits gained by an honest super-agent and a dishonest super-agent respectively. As shown in Figure 5.8, honest super-agents can gain many more virtual credits than dishonest super-agents. Dishonest super-agents do not have much chance to be asked by consumer agents for advice about service providers and cannot gain many virtual credits. Therefore, it is better off for super-agents to provide truthful community-based reputation.

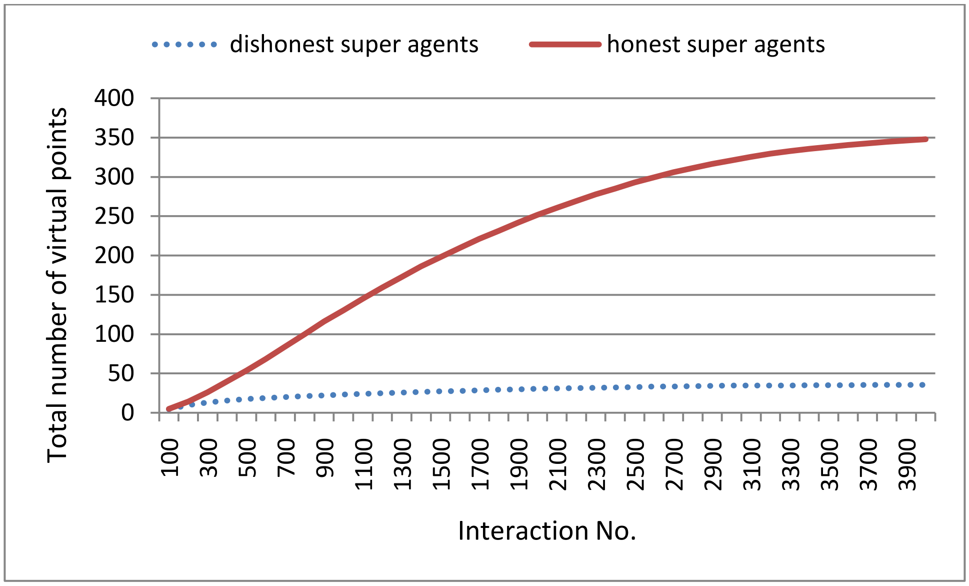


Figure 5.8 Incentives for super-agents to be honest

In the third experiment, I show that forming communities promotes the honesty of consumer agents. A dishonest consumer agent is randomly selected from each group of consumer agents (i.e. picky, mid-picky and non-picky). When a consumer agent acts dishonestly, it will always provide opposite feedback to super-agents. For example, if its feedback rating is 1, it will say it is 0, vice versa. I compare the successful interaction ratio between the dishonest consumer agent with the rest of consumer agents in the same group. Figure 5.9 shows that it is not beneficial for consumer agents to act dishonestly. When a consumer agent provides false feedback, it has a higher chance to join a wrong community (one that does not match its preferences or quality criteria) and to be excluded from the right community. Therefore, it will lose valuable information from the right communities, no matter whether it is non-picky, middle-picky or picky.

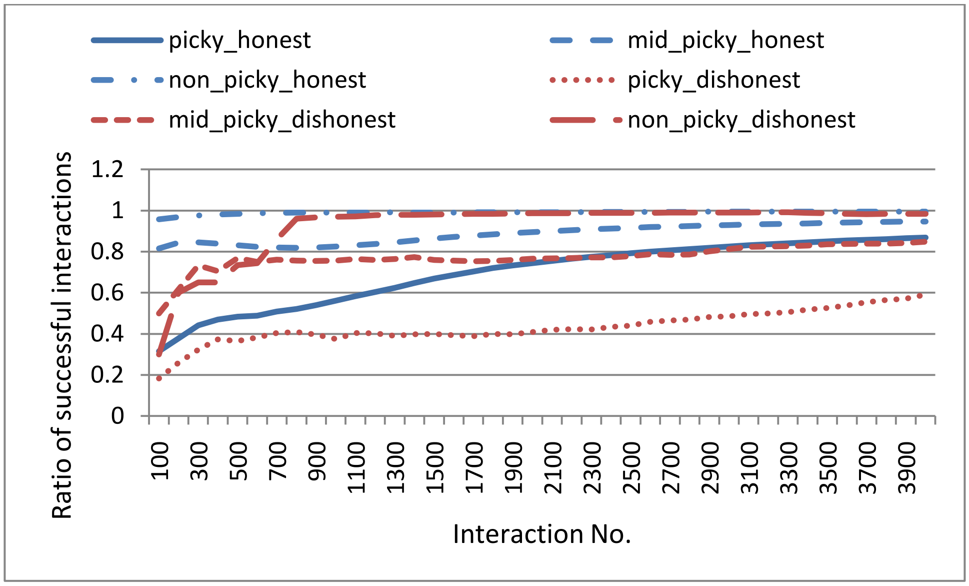


Figure 5.9 Incentives for consumer agents to be honest

5.3.3 Comparative Results

I finally carry out a set of experiments to compare my community-based approach with the experience-based approach [59] and the model of Yu and Singh [86]. The experience based approach allows consumer agents to share experience with services expressed using pre-defined ontologies, in order to cope with subjective differences among consumers. The model of Yu and Singh relies on consumer agents themselves to model other consumers and form implicit communities. Let's see an example to show the difference between the two methods.

Suppose there is a service providing file downloading and consumer agents will evaluate the service only based on the download speed it provides. When a consumer agent *CA1* uses the service, it gets a download speed of 50 kb/second. The agent *CA1* thinks the download speed is OK and gives a satisfying evaluation. Later, another consumer agent *CA2* asks *CA1*for its advice about the service. In the experience based method, agents will share the facts about their experiences with services, not their judgments. Therefore, *CA1* will tell *CA2* that it used the service and the service's download speed is 50 kb/second. Then *CA2* will judge the service according to its own judging criteria. Since agent *CA2* is very picky, it will evaluate the service as unsatisfying since it think the download speed of 50 kb/second is too slow and the service is not good to use. However, in Yu and Singh's method, agents share their evaluations about services. *CA1* will tell *CA2* that the service is satisfying. But after *CA2* uses the service, it thinks it unsatisfying, even although it got the same download speed as *CA1*. Because of the mismatch of their evaluations about the service, *CA2* can reduce its trust in *CA1*, which indicates the difference between their judging criteria. Next time, *CA2* will not choose *CA1* to ask for advice of services.

The results shown in Figure 5.10, Figure 5.11 and Figure 5.12 indicate that the community-based approach outperforms the experience-based approach in the beginning when consumer agents do not have many interactions with services. Later on when consumers have a larger number of interactions with services, these two approaches produce similar results. These results confirm that my community-based approach can effectively cope with subjective differences of consumers but requires no effort to define ontologies at design time.

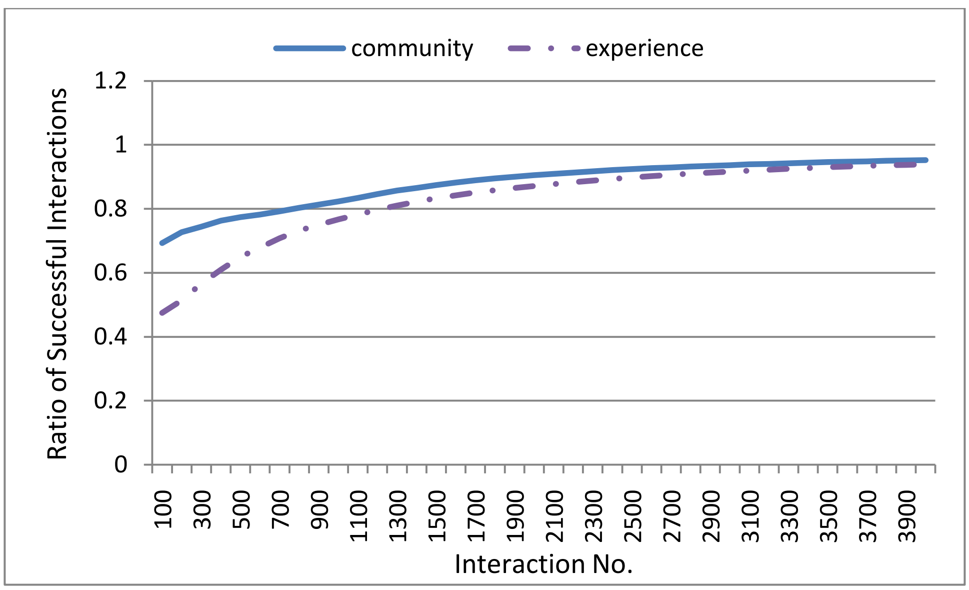


Figure 5.10 Community-based versus experience-based (the mean)

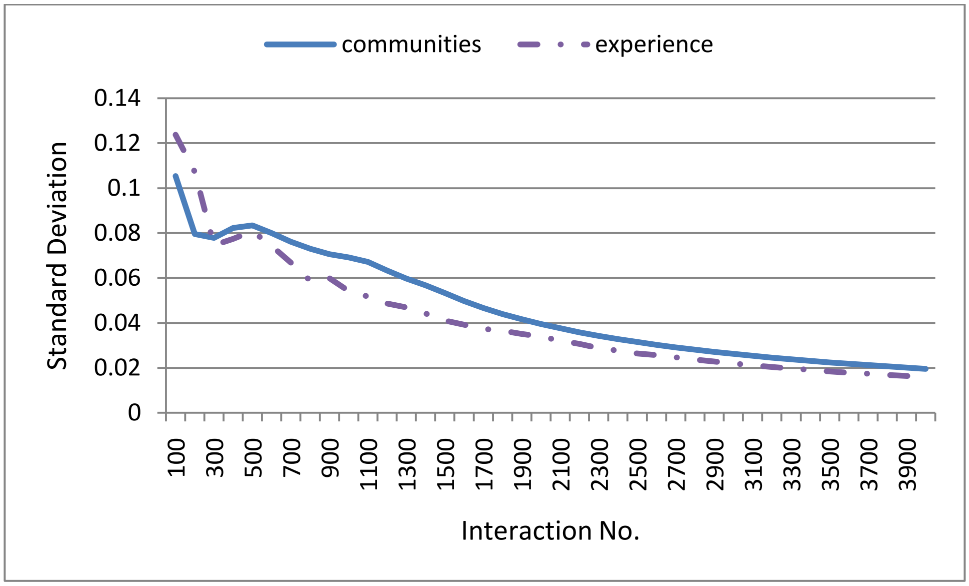


Figure 5.11 Community-based versus experience-based (the standard deviation)

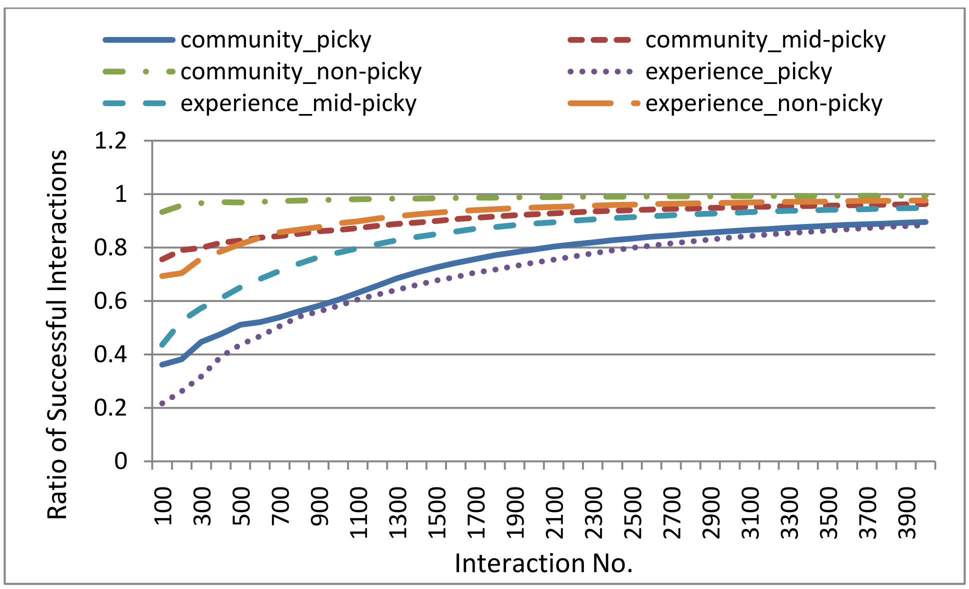


Figure 5.12. Community-based versus experience-based for different groups

The results shown in Figure 5.13, Figure 5.14 and Figure 5.15 indicate that the community-based approach outperforms Yu and Singh's model in the beginning when consumer agents do not have many interactions with services. Later on when consumers have more interactions with services, my approach is slightly worse than Yu and Singh's model. However, when consumers have a larger number of interactions (i.e. more than 3000) with services, these two approaches produce similar results. These differences between the two approaches can be interpreted as follows. In Yu and Singh's model, a consumer agent relies on its own experiences to build its neighborhood list in order to form implicit communities. Since a consumer agent does not have much experience with services in the beginning, it cannot form its community quickly. However, my approach makes use of super-agents to form communities. They can quickly accumulate experiences from multiple consumer agents and build effective communities from the beginning. Therefore at the beginning, my approach performs better than Yu and Singh's model. Later on, when consumer agents have more experience with services, Yu and Singh's model performs a little better. Because in their model, consumer agents rely only on their personal experience to create communities, these communities are more personalized and can help the consumers find more satisfactory services. However, after consumers have enough personal experience with services, they do not rely on other consumers' opinions or community-based reputation information about services. In this case, the performance of my community-based approach is similar to that of Yu and Singh's model. Another important point is that the model of Yu and Singh also requires much effort from consumer agents to model many other consumers. Comparably, in my approach, consumers only need to model super-agents managing the communities that they belong to, thus minimizing the effort required from consumers.

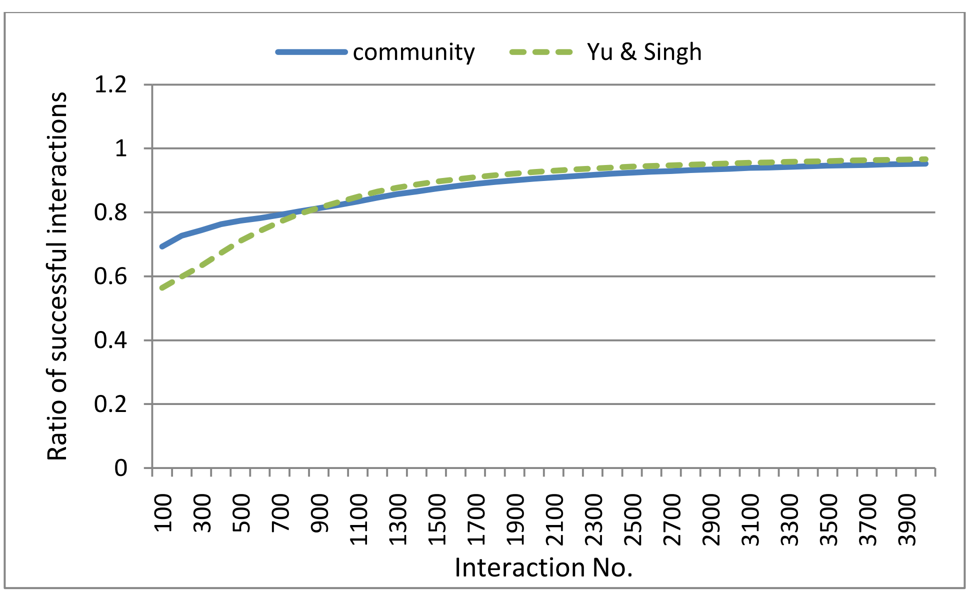


Figure 5.13. Community-based versus Yu and Singh's model (the mean)

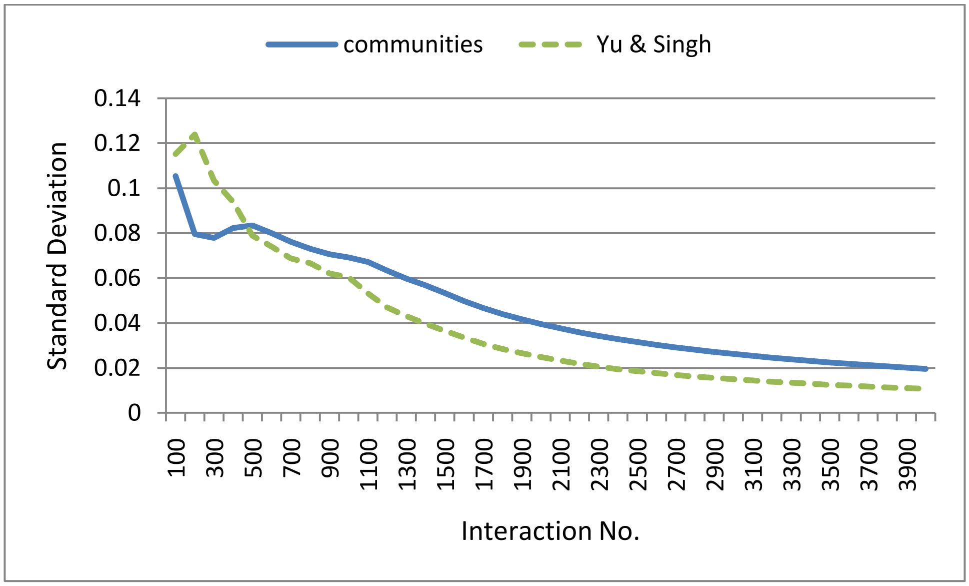


Figure 5.14. Community-based versus Yu and Singh's model (the standard deviation)

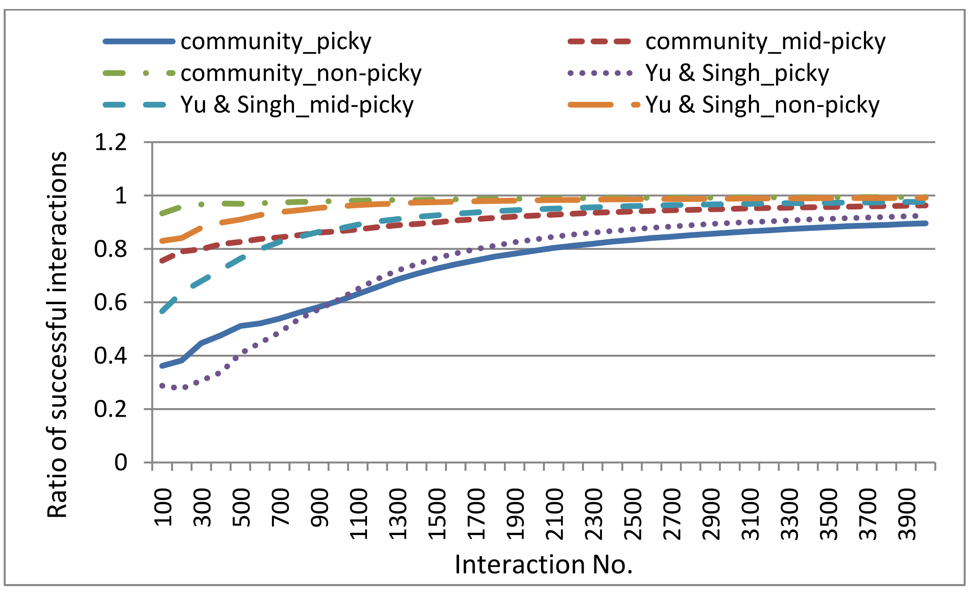


Figure 5.15 Community-based versus Yu and Singh's model

Table 5-4 shows a summary of the comparison of my approach with other approaches. Compared with the experienced-based model [59], my approach needs less effort at the design stage, since it does not use ontologies to deal with subjective differences among agents. Defining ontologies cost time and effort and they need to be changed for different applications. Experiments also show that my approach performs better than the experienced-based model.

My approach has advantages over Yu and Singh's model [86] which is considered a benchmark for decentralized trust and reputation systems. In Yu and Singh's model, agents need to model each other, which adds burden to the agents and requires them to do more computation and communication. Therefore, it is less suitable for a network where there are many agents with poor capabilities (e.g. CPU power and bandwidth), such as a mobile (cell phone) network. In contrast, my approach allows super-agents to take most responsibilities, so that agents with poor capabilities can benefit from the super-agents. In addition, the experiments also show that at the beginning, my approach can form communities more effectively than Yu and Singh's method. This brings advantage for new agents who have not yet accumulated experience. They can take advantage of the existing communities built by the super-agents.

Table 5-4 Comparison of different approaches

	Community-based model	Yu and Singh's model	Experienced-based model
Community	explicit	implicit	no
Ontology	no	no	yes
Modeling other consumer agents	no	yes	no
Performance at the start (for new agents)	best	better	good
Performance in a long run	better	best	good

5.4 Discussion

According to the typology defined in Section 2.3, trust and reputation mechanisms can be classified either as global or as personalized/community-based. In global reputation systems, the reputation of an entity (i.e. a person/agent/product/service) is based on the opinions from the general population, which is public and visible to all the consumer agents. In personalized/community-based reputation systems, for a requesting consumer, the reputation of an entity is based on the opinions either from a group of consumer agents chosen by the requesting consumer agent (in this case it is called personalized reputation), or from the members of a community, formed by some other criterion (in this case it is called community-based reputation). Most of trust and reputation mechanisms can only build one kind of reputation, either global reputation or personalized/community-based reputation.

The methods proposed in Chapter 4 and Chapter 5 together can support both of kinds of reputation. A global reputation is usually based on a larger population. It can be built faster than a personalized/community-based reputation. Before a consumer agent can obtain a personalized/community-based reputation, using a global reputation can help a consumer make a fairly good selection. But compared with a personalized/community-based reputation, a global reputation is less effective since for a requesting consumer, a personalized/community-based reputation is built on the opinions of a group of trustworthy consumers, which are more likely to have similar interests and judging criteria. Therefore, when a personalized/community-based reputation is available, it is better to use it instead of a global reputation.

My method of using super-agents to build reputation and form communities brings the advantages of efficiency from centralized systems to decentralized systems. Building global reputation is easy and efficient in a centralized system, but hard and inefficient in a decentralized system. Using super-agents makes it easier and more efficient to build global reputation of services in decentralized systems compared to other methods, such as Eigentrust [28], PeerTrust [84], P-Grid [2]. The drawback of these methods is that they are all designed for a structured network based on DHT (distributed hash table) or P-Grid. A structured network is hard to cope with the situation where agents arbitrarily join and leave. In Eigentrust [28], every agent has a reputation manager determined by a hash function. The reputation manager is responsible for computing the global reputation for an agent based on the algorithm of power iteration. This method is very complex and also requires the availability of related agents when calculating the global reputation of an agent. PeerTrust [84] is similar to Eigentrust. Moreover, it introduces several different parameters and factors that can be used to calculate the reputation of agents, and a method of how to combine them. Aberer and Despotovic [2] proposed a complaint-based reputation system. A specially designed P-Grid structure is used to determine reputation managers for agents. An agent’s reputation managers are responsible for collecting other agents’ complaints about the agent. If there are more complaints against an agent, the agent's reputation will be worse. All the above mentioned approaches require a lot of computation and communication between agents and cannot be used in an unstructured network, like Gnutella networks or super-peer networks. My method can be applied to unstructured networks. It is more intuitive and more efficient in term of computation and communication cost. The super-agent building reputation for a service acts like a central node for the service. It collects feedback about the service from all consumer agents. It provides consumer agents with the global reputation of the service when requested. Super-agents can also form various communities to bring together agents with similar interests and judging criteria. The community can serve as an information space for agents with similar interests and judging criteria to share their information about services and their opinions about services. In addition, my method does not have the problem of single point of failure or scalability problems typical for centralized systems, since it naturally assumes redundancy; multiple super-agents may be responsible for managing the reputation of the same service or build multiple similar communities. To some extent, having multiple similar communities can prevent the problem of "groupthink" [79], which may occur in a cohesive group where its members can reach consensus to make faulty decisions because these members are very similar in background and this group is isolated from outside opinions. In the case of communities formed by super agents, community members can engage in groupthink by only selecting a few of services recommended by their communities and ignoring other good services or new services outside their communities. In order to overcome this problem, community members can occasionally ask other agents outside their communities for their opinions on services and try the services recommended by them.

5.5 Summary

In this chapter, I propose a community-based approach for service selection where super-agents with more capabilities serve as community managers. They maintain communities and build community-based reputation for a service based on the opinions from all community members that have similar interests and judgment criteria. The community-based reputation is useful for consumer agents in selecting satisfactory services when they do not have much personal experience with the services. Experimental results show that this approach can help agents find good services and have more successful interaction. A practical reward mechanism is also introduced to create incentives for super-agents to contribute their resources and provide truthful community-based reputation information, as strong support for the approach.

In summary, my work has several unique features. First, forming explicit communities brings benefits in service selection, including more valuable information shared by like minded agents in communities and easier finding of reputation data through communities. Second, the proposed practical reward mechanism encourages incentives for super-agents to contribute their resources, form communities, and truthfully share their reputation information.

Chapter 6

Conclusions and Future Work

This chapter gives a summary of the research contributions and points out some future directions.

6.1 Contributions

In a nutshell, the purpose of this research is to promote effective, efficient and flexible trust and reputation management in decentralized systems. The super-agent based mechanisms also enable a self-organizing system. It lets the agents with more capabilities (i.e. super agents) take more responsibilities to organize the information and other agents thus creating a decentralized system that has local centers. The resulting system has the flexibility of a decentralized system, but is more efficient.

The contributions of this research can be summarized in the following aspects:

- I proposed a Bayesian network-based trust model. Trust is context-dependent and multi-faceted. Agents need to develop differentiated trust and learn reputation in different aspects of other agents' capability. The agent's needs are different in different situations. Depending on the situation, an agent may need to consider its trust in a specific aspect of another agent's capability or in multiple aspects. My Bayesian network based trust model provides a flexible method to present differentiated trust and combines different aspects of trust to meet agents' different needs.
- I proposed a mechanism of using super-agents to manage reputation. Super-agents can build general public opinion-based reputation and provide reputation information to other agents, while other agents can help super-agents build reputation by sharing their ratings with the super-agents. This mechanism will help agents find reputation information efficiently and effectively.
- I proposed a mechanism for super-agents to form communities. Super-agents can build their communities to bring together agents with similar interests and judging criteria. A community can provide its community members with community-based reputation of services that is personalized for the members' needs and can facilitate them make

better selection of services.

- A reward mechanism is designed to encourage super-agents to contribute their resources and share the reputation information they build, and also to be truthful. Super-agents get rewards for their work and honest behavior. They are therefore motivated to use their resources for the benefit of the community.
- The above mechanisms together create a flexible unified framework to enable efficient and effective trust and reputation management in decentralized systems. The Bayesian network-based trust model enables agents to develop differentiated trust in other agents. The super-agent based approach facilitates agents to discover reputation information efficiently and effectively. The idea of allowing super-agents to form communities and to share reputation information further enables personalized reputation management among agents with similar interests and judging criteria. The reward mechanism proposed in the framework is then a complementary element, which creates incentives for participation and honesty. As more and more agents contribute their resources and act honestly, my framework is able to create a better environment for the agents to find appropriate services and information in decentralized networks.

6.2 Other Potential Applications

Although the mechanisms proposed in this thesis focus on decentralized systems, such as file sharing systems and decentralized web service systems, some of the mechanisms can also be applied to centralized systems, e.g. e-commerce websites.

The idea of using Bayesian networks to model trust can be used to model differentiated reputation in centralized systems. Figure 3.8 in Chapter 3 has shown a simple example of how the Bayesian network based method can be applied in eBay to model users' differentiated reputation. Besides the area of trust and reputation, this model may be applicable to other areas where there is a need to model differentiated aspects of an entity (e.g. a person/agent, a product, or an event). For example, in the area of user modeling, this model also bears some resemblance with the work on distributed user modeling and purpose-based user modeling [45][62].

The idea of using super-agents to form interest-based communities can be applied in the systems where users have different interests and judging criteria. Users that have a strong interest in some specific aspect or users with a lot of knowledge can act as super-agents to create their communities. The communities can serve as a space for users to share and look for information. For example, in Amazon [73], a user wants to buy a book. He does not know which book he should look for. Instead of looking for a specific book, he searches for the book community that he is a member of. In the community, he can browse the books recommended by the community, access the members' evaluations on the books, and finally choose one to buy. This method can also be used for a digital library where readers can form interest-based communities to share their reviews and recommend books to each other. Some other areas that this method could be applied to are education (lifelong learning), health care, gaming and so on.

6.3 Future Work

There are several directions for future work that are worth exploring. Work along some of these directions has already started.

6.3.1 Adding a Time Decay Factor into Bayesian Networks

In the Bayesian network based trust model, peers use naïve Bayesian networks to model their trust in file providers. A peer will develop its naïve Bayesian network based on its past interactions with a file provider. In the Bayesian network based trust model, new interactions are treated equally as the old ones. However, as mentioned in Section 1.1.1, trust is dynamic and new experiences should have more importance than old ones, since old experiences may become obsolete or irrelevant with time passing by. When a peer models a file provider, recent experiences are more important than old ones, since a file provider may change its behavior over time. For example, a file provider may be good at the beginning by providing fast downloading speed and good-quality files. Later on, it may change to be a bad file provider with slow downloading speed and bad-quality files. Therefore, past experiences have to be discounted, so that peers can quickly adapt to the change of file providers' behaviors.

In order to make the Bayesian network based trust model adapt to changes in file providers' behavior, when a peer develops its trust model for a file provider, it will maintain a table for each node in the Bayesian network, which includes the conditional probability and the number of interactions given the corresponding condition. Table 6-1 shows an example of the CPT tables for the root node “T” in Figure 3.3. In the table, *a* and *b* denote the total number of interactions when they are “satisfying” and “not satisfying” respectively. The conditional probability is calculated based on these interaction numbers. The technique of exponential smoothing [78] is used to discount the past experiences in Bayesian networks. For each time interval, the number of interactions in each table will be discounted by multiplying it by a time decay factor α, a value between 0 and 1. The lower the value of α, the faster the past experiences will be forgotten. The new experiences will become more important. For example, a will be replaced with $a * \alpha$. Suppose α equals 0.9. Table 6-2 shows the updated CPT tables for node “T” after one time interval. For example, the number of satisfying interactions was 10 in Table 6-1, and now it is decreased to 9. When a peer gains new experiences with a file provider in the current time period, the new experiences will be added to the discounted past experiences. Suppose there are two new successful interactions. Table 6-3 shows the number of satisfying interactions increased from 9 to 11.

Table 6-1 The CPT table for node “T”

Conditions	The number of interactions given the condition	Probabilities
T = 1	a = 10	P(T=1) = a / (a + b)
T = 0	b = 2	P(T=0) = b / (a + b)

Table 6-2 The CPT table for node “T” with time decay

Conditions	The number of interactions given the condition	Probabilities
T = 1	**a = 10 * 0.9 = 9**	P(T=1) = a / (a + b)
T = 0	b = 2 * 0.9 = 1.8	P(T=0) = b / (a + b)

Table 6-3 The CPT table for node "T" with new experiences

Conditions	The number of interactions given the condition	Probabilities
T = 1	***m = 9 + 2 = 11***	P(T=1) = m / (m + b)
T = 0	b = 1.8	P(T=0) = b / (m + b)

For future work, some experiments will be carried out to see whether this modified Bayesian network trust model will help peers build a better model of file providers with a dynamic behavior.

6.3.2 Personalization

In the community formation mechanism, a super-agent, the community manager, is responsible for building and providing a collective view of services' reputation based on its members' ratings. This decision has certain disadvantages, such as the information loss in the aggregation process, as explained below.

Information loss happens always when there is an aggregation of opinions, i.e. in any form of centralized reputation system. A community creates a form of centralization, though it is still tailored to specific group of agents who share certain interests. Yet, in a community, an agent may trust its members differently, i.e. it may trust some members more and others less. An aggregated view of the "goodness" of a service, computed by using the information provided by the entire community may not be good for an individual agent to make a decision. It does not assign more weight to the opinions from the community members that the individual agent trusts the most. In addition, the individual agent does not know how the collective reputation is calculated by the community manager and has to take the overall reputation value "with a grain of salt" depending on how much it trusts the community as a whole. In addition, since the super-agent (community manager) has full control of handling the community members' ratings, it can easily provide fake reputation information to other agents in order to badmouth or boost up some services.

For future work, the community formation mechanism can be extended by allowing the community manager to provide its members' original ratings. An agent can decide how to handle these ratings, combine them, and get their own personalized representation of the service reputation, which will be closer to the agent's own opinion than the reputation generated by the community manager.

6.3.3 Other Future Work Directions

There are some other directions for future work. I will try different simulation environments to test my approaches, such as a simulated P2P system. I will also change the parameters used in my simulation, for example, the percentage of super-agents or the percentage of service providers, to see how these changes affect the system's performance. I will test different scenarios when super-agents are not reliable. For example, let them lose information or be unavailable.

For future work, an analysis of the robustness of the proposed reputation system with respect to attacks would be an interesting undertaking. There are various possible attacks [23] on reputation systems. For example, the *Sybil attack* where an attacker can create multiple fake identities to boost or degrade other agents' reputation, *the whitewash attack* where an attacker can regain its reputation after malicious behaviors and continue to behave badly, and *the collusion attack* where a group of malicious agents can work together to artificially manipulate other agents' reputation. Various modifications of the mechanisms are possible to ensure robustness against attacks, for example, allowing consumer agents to develop trust in super-agents by asking advice from other consumer agents. The honesty of the other consumers in providing information about the trustworthiness of super-agents may also need to be modeled. As the current reward mechanism is rather simple, I will analyze deeply the possible strategies of service providers in offering rewards. For example, a service provider, especially a new one, may offer higher rewards than an established service provider to attract super-agents to build its reputation in order to gain more consumers. I will also look into the idea of sharing information about community members among different communities by super-agents (community managers). This will be helpful to effectively grow communities [31].

6.4 Concluding Remarks

The purpose of this research is to promote effective, efficient and flexible trust and reputation management in decentralized systems. In this thesis, I propose several ideas to achieve this purpose, including the Bayesian network based trust modeling, the idea of using super-agents to manage reputation and forming interest based communities, and using rewards to encourage super-agents contributing their resources and being honest. Experiments confirm that these approaches can achieve better effectiveness, efficiency and flexibility compared to the systems that do not use them. However, these methods are not perfect, there is need to do more improvements so that these approaches can be applicable to real-world environments.

References

[1] Aberer K., "P-Grid: A Self-Organizing Access Structure for P2P Information Systems". Proceedings of the Ninth International Conference on Cooperative Information Systems, 2001, pp. 179-194.

[2] Aberer K. and Despotovic Z., “Managing Trust in a P2P Information System”. Proceedings of ACM CIKM International Conference on Information and Knowledge Management, 2001, pp. 310-317.

[3] Abdul-Rahman A. and Hailes S., “Supporting Trust in Virtual Communities”. Proceedings of the Hawai’i International Conference on System Sciences, Maui, Hawaii, Jan 4-7 2000.

[4] Bailey B., Gurak L., Konstan J., "Do You Trust Me? An Examination of Trust in Computer-Mediated Exchange." Human Factors and Web Development, 2nd edition. Mahwah, NJ: Lawrence Erlbaum, 2002.

[5] Barabási A., "Linked: the New Science of Networks". Perseus Publishing, Hard cover, 229 pgs, April 2002.

[6] Beenen G., Ling K., Wang X., Chang K., Frankowski D., Resnick P., and Kraut R. E., “Using Social Psychology to Motivate Contributions to Online Communities”. Proceedings of CSCW2004, Chicago, IL, November 2004, pp. 212-221.

[7] Breban S., Vassileva J., “A Coalition Formation Mechanism Based on Inter-Agent Trust Relationships”. Proceedings of the First Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Part 1, ACM Press, 2002, pp. 306-308.

[8] Breese, J. S., Heckerman, D. and Kadie, C., “Empirical Analysis of Predictive Algorithms for Collaborative Filtering”. Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI98), 1998, pp. 43-52.

[9] Bretzke H., Vassileva J., “Motivating Cooperation in Peer to Peer Networks”. Proceedings User Modeling UM03, Johnstown, PA, Springer Verlag LNCS 2702, June 22-26, 2003, pp. 218-227.

[10] Brooks H, C. and Durfee. H. E., “Congregating and Market Formation”. Proceedings of the first international joint conference on Autonomous agents and multiagent systems (AAMAS), Bologna, Italy, 2002, pp. 96-103.

[11] Castelfranchi, C. and Falcone, R., “Principles of Trust for MAS: Cognitive Anatomy, Social Importance, and Quantification”. Proceedings of Third International Conference on Multiagent Systems, 1998, pp. 72-79.

[12] Castelfranchi C., Falcone R., Pezzulo G., “Trust in Information Sources as a Source for Trust: a Fuzzy Approach”. AAMAS 2003, pp. 89-96.

[13] Cheng R. and Vassileva, J., “User Motivation and Persuasion Strategy for P2P Communities”. Proceedings HICSS'2005 (Mini-track on Online Communities in the Digital Economy/Emerging Technologies), Hawaii, Jan 3-6, 2005, pp. 193a.

[14] Conitzer V. and Sandholm T., “Complexity of Constructing Solutions in the Core Based on Synergies among Coalitions”. Artificial Intelligence, 170(6-7), 2006, pp. 607-619.

[15] Cornelli F. and Damiani E., “Implementing a Reputation-Aware Gnutella Servant”. Proceedings of the International Workshop on P2P Computing, Pisa, Italy, May 24, 2002.

[16] Damiani E., Vimercati D., Paraboschi S., Samarati P. and Violante F., “A Reputation-Based Approach for Choosing Reliable Resources in P2P Networks”. Proceedings of the 9th ACM conference on Computer and Communications Security (CCS’02), ACM, 2002, pp. 207–216.

[17] Dellarocas C., “Immunizing Online Reputation Reporting Systems Against Unfair Ratings and Discriminatory Behavior”. Proceedings of 2nd ACM Conference on Electronic Commerce, 2000, pp. 150-157.

[18] Dellarocas C. and Resnick P., “Online Reputation Mechanisms: a Roadmap for Future Research”. Summary report of the First Interdisciplinary Symposium on Online Reputation Mechanisms, April 26-27, 2003.

[19] Flake,W. G., Lawrence, S., Giles, C. L. & Coetzee, F., “Self-Organization and Identification of Communities”. IEEE Computer 35, 2002, pp. 66 – 71.

[20] Gwebson D., Kleinberg J., and Raghavan P., “Inferring Web Communities from Link Topology”. Proceedings of the 9th ACM conference on Hypertext and Hypermedia, 1998, pp. 225-234.

[21] Good N., Schafer J., Konstan J., Borchers A., Sarwar B., Herlocker J. and Riedl J., "Combining Collaborative Filtering with Personal Agents for Better Recommendations". Proceedings of the 1999 National Conference of the American Association of Artificial Intelligence, pp 439-446.

[22] Hales, D., “Group Reputation Supports Beneficent Norms”. The Journal of Artificial Societies and Social Simulation (JASSS) vol. 5, no. 4, 2002

[23] Hoffman K, Zage D., and Nita-Rotaru C., "A Survey of Attack and Defense Techniques for Reputation Systems". ACM Computing Surveys, Volume 41, Issue 4, December 2009.

[24] Indratmo and Vassileva J., "Social Interaction History: a Framework for Supporting Exploration of Social Information Spaces". Proceedings of the IEEE International Symposium on Social Computing Application, Vancouver, Canada 2009, pp. 538—545.

[25] Jordan M., “Learning in Graphical Models”, MIT Press, 1998.

[26] Jøsang A., Ismail R., and Boyd C., “A Survey of Trust and Reputation Systems for Online Service Provision”. Decision Support Systems, Volume 43, Issue 2, 2007, pp.618-644.

[27] Jovanovic M., “Modeling Large-Scale P2P Networks and a Case Study of Gnutella”, University of Cincinnati, master thesis, April 2001.

[28] Kamvar S. D., Schlosser M. T., and Garcia-Molina H., “The Eigen Trust Algorithm for Reputation Management in P2P Networks”. Proceedings of the twelfth International Conference on World Wide Web, 2003, pp. 640-651.

[29] Karame G., Christou T. I., and Dimitriou T., “A Secure Hybrid Reputation Management System for Super-Peer Networks,” Proceedings of the Consumer Communications and Networking Conference, 2008, pp. 495–499.

[30] Fullam K., Klos T., Muller G., Sabater J., Schlosser A., Topol Z., Barber K., Rosenschein J., Vercouter L., Voss M., “A Specification of the Agent Reputation and Trust (ART) Testbed: Experimentation and Competition for Trust in Agent Societies”. Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems, Utrecht, The Netherlands, July 2005, pp. 512-518.

[31] Kastidou G., Larson K., and Cohen R., “Exchanging Reputation Information between Communities: A Payment-Function Approach,” Proceedings of the International Joint Conference on Artificial Intelligence, 2009, pp. 195-200.

[32] Khambatti M., Dasgupta P. and Ryu D.K., “A Role-Based Trust Model for P2P Communities and Dynamic Coalitions”. IEEE International Information Assurance Workshop (IWIA), Charlotte NC, April 2004, pp. 141.

[33] Li C., Sycara K., “Algorithm for Combinatorial Coalition Formation and Payoff Division in an Electronic Marketplace”. Proceedings of AAMAS, 2002, pp. 120-127.

[34] Macnaughton-Smith P., Williams, W.T., Dale, M.B., and Mockett, L.G., “Dissimilarity Analysis: a New Technique of Hierarchical Sub-Division”. Nature (202), 1964, pp. 1034-35.

[35] Marsella, C. S., Adwebi, J., Al-Onaizan, Y., Kaminka, G. A., Muslea, I., Tallis, M., and Tambe, M., “On Being a Teammate: Experiences Acquired in the Design of RoboCup Teams”. Proceedings of the third annual conference on Autonomous Agents, 1999, pp. 221-227.

[36] Marsh S., “Formalising Trust as a Computational Concept”. Ph.D. Thesis, University of Stirling, 1994.

[37] Marti S. and Garcia-Molina H., “Examining Metrics for P2P Reputation Systems” Technical Report 2003-39, Stanford University, 2003.

[38] Maximilien E. and Singh M., “Toward Autonomic Web Services Trust and Selection”. Proceedings of the 2nd international conference on Service oriented computing, ACM press, New York, NY , USA, 2004, pp.212-221.

[39] McKnight, D. H., & Chervany, N. L., “Conceptualizing Trust: A Typology and E-Commerce Customer Relationships Model”. Proceedings of the 34th Hawaii International Conference on System Sciences, 2001.

[40] Melo F. and Veloso M., “Learning of Coordination: Exploiting Sparse Interactions in Multiagent Systems”. AAMAS (2) 2009, pp.773-780.

[41] Modarresi A., Mamat A., Webrahim H., and Mustapha N., “A Community-Based P2P Model Based on Social Networks”. International Journal of Computer Science and Network Security 8(4), 2008, pp. 272-277.

[42] Montaner M., López B. and Rose J., “Developing Trust in Recommender Agents”. Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'02), Palazzo Re Enzo (Italy), 2002, pp. 304-305.

[43] Mui L., Halberstadt A. and Mohtashemi M., “Notions of Reputation in Multi-Agents Systems: A Review”. Proceedings of Autonomous Agents & Multiagent Systems (AAMAS’02), Bologna, Italy, 2002, pp. 280–287.

[44] Nair, R., Tambe, M., and Marsella, S., “Role Allocation and Reallocation in Multiagent Teams: Towards a Practical Analysis”. Proceedings of the Second International Joint

Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2003, pp. 552-559.

[45] Niu X., McCalla G., Vassileva J., “Purpose-Based User Modelling in a Multi-Peer Portfolio Management System”. Proceedings of User Modeling UM03, Johnstown, PA, June 22-26, 2003, pp. 398-402.

[46] Page L., Brin S., Motwani R., and Winograd T., "The PageRank Citation Ranking: Bringing Order to the Web". Technical report, Stanford Digital Library Technologies Project, 1998.

[47] Paolo M. and Bobby B., “Using Trust in Recommender Systems: an Experimental Analysis”. Second International Conference on Trust Management, Oxford, England, 2004, pp. 221-235.

[48] Pujol M., Sanguesa R., and Delgado J., “Extracting Reputation in Multi Agent Systems by Means of Social Network Topology”, Proceedings of Int'l Conf. on Autonomous Agents and Multi-Agents Systems (AAMAS-02), 2002, pp. 467-474.

[49] Regan K., Poupart P., and Cohen R., “Bayesian Reputation Modeling in E-Marketplaces Sensitive to Subjectivity, Deception and Change”. Proceedings of the Twenty-First Conference on Artificial Intelligence (AAAI), 2006, pp. 1206-1212.

[50] Resnick P. and Zeckhauser R., “Trust among Strangers in Internet Transactions: Empirical Analysis of eBay’s Reputation System”. NBER Workshop on Empirical Studies of Electronic Commerce, 2000.

[51] Rodriguez-Perez M., Esparza O., and Munoz L. J., “Surework: a Super-peer Reputation Framework for P2P Networks”. Proceedings of ACM Symposium on Applied Computing, 2008, pp. 2019-2023.

[52] Sabater, J. and Sierra, C., “Reputation and Social Network Analysis in Multi-Agent Systems”. Proceedings of International Conference on Autonomous Agents and Multi-Agents Systems (AAMAS-02), 2002, pp. 475-482.

[53] Sabater J. and Sierra C., “REGRET: Reputation in Gregarious Societies”. Proceedings of the fifth international conference on Autonomous agents, 2001, pp. 194-195.

[54] Sabater J., “Trust and Reputation for Agent Societies”. PhD thesis, Institute for Artificial Intelligence Research, Bellaterra, 2003.

[55] Saroiu S., Gummadi P., Grwebble S., “A Measurement Study of P2P File Sharing Systems”. Proceedings of Multimedia Computing and Networking (MMCN), 2002.

[56] Schafer J., Konstan J.and Riedl J., “Recommender Systems in E-commerce”. Proceedings of the 1st ACM conference on Electronic commerce, Denver, Colorado, United States, November 3-5, 1999, pp. 158-166.

[57] Schillo, M., Funk, P., and Rovatsos, 2000., “Using Trust for Detecting Deceitful Agents in Artificial Societies”. Applied Artificial Intelligence, vol. 14, pp. 825-848.

[58] Sen, S. and Sajja, N., “Robustness of Reputation-Based Trust: Boolean Case”, Proceedings of Int'l Conf. on Autonomous Agents and Multi-Agents System (AAMAS-02), 2002, pp. 288-293.

[59] Sensoy M., Zhang J., Yolum P., and Cohen R., “Poyraz: Context- Aware Service Selection under Deception”. Computational Intelligence, vol. 25, no. 4, 2009, pp. 335–366.

[60] Shehory O. and Kraus S., “Methods for Task Allocation via Agent Coalition Formation”. Artificial Intelligence Journal, Vol. 101 (1-2), May 1998, pp 165-200.

[61] Tang T.Y, Winoto P. and Niu X., “Who can I trust? Investigating Trust between Users and Agents in a Multi-Agent Portfolio Management System”. AAAI-2002 Workshop on Autonomy, Delegation, and Control: From Inter-agent to Groups. Edmonton, Canada.

[62] Vassileva J., McCalla G. and Greer J., “Multi-Peer Multi-User Modeling”. User Modeling and User-Adapted Interaction, 13:(1), 2003, pp. 179-210.

[63] Vassileva, J., “Motivating Participation in Peer to Peer Communities”. Proceedings of the 3rd international conference on Engineering societies in the agents world III, ESAW'02, Madrid, 16-17 September 2002, pp. 141-155.

[64] Vassileva J., Breban S. and Horsch M., “Agent Reasoning Mechanism for Long-Term Coalitions Based on Decision Making and Trust”. Computational Intelligence, Vol. 18, no. 4, 2002.

[65] Vu L., Hauswirth M., Aberer K., “QoS-Based Service Selection and Ranking with Trust and Reputation Management”. Proceedings of OTM'05, R. Meersman and Z. Tari (Eds.), LNCS 3760, 2005, pp. 466-483.

[66] Wang Y. and Vassileva J., “Trust and Reputation Model in P2P Networks”. Proceedings of The Third IEEE International Conference on P2P Computing, Linköping, Sweden, September 1-3, 2003.

[67] Wang Y. and Vassileva J., “Trust-Based Community Formation in P2P File Sharing Networks”. Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence (WI 2004), Beijing, China, September 20-24, 2004, pp. 341-348.

[68] Wang Y. and Vassileva J., “Toward Trust and Reputation Based Web Service Selection: a Survey”. International Transactions on Systems Science and Applications (ITSSA) Journal, special Issue on New tendencies on Web Services and Multi-agent Systems (WS-MAS), Vol 3, No. 2, 2007.

[69] Wang Y. and Vassileva J., “Super-Consumer Based Reputation Management for Web Service Systems”. Proceedings the 2nd IEEE International Conference on Digital Ecosystems and Technologies, IEEE-DEST 2008, Phitsanulok, Thailand, 26-28 February, 2008, pp. 444-451.

[70] Wang Y., Zhang J., Vassileva J., "Super-Agent Based Reputation Management with a Practical Reward Mechanism in Decentralized Systems". Proceedings of the IFIP WG 11.11 International Conference on Trust Management, 2010.

[71] Wang Y., Zhang J., and Vassileva J., "Effective Web Service Selection via Communities Formed by Super-Agents". Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI), 2010, pp. 444-451.

[72] Whitby A., Jøsang A. and Indulska J., “Filtering Out Unfair Ratings in Bayesian Reputation Systems”. The Journal of Management Research, 4(2), February 2005, pp.48-64.

[73] www.amazon.com

[74] www.ebay.com

[75] www1.epinions.com

[76] www.google.com

[77] www.movielens.org

[78] http://en.wikipedia.org/wiki/Exponential_smoothing

[79] http://en.wikipedia.org/wiki/Groupthink

[80] http://en.wikipedia.org/wiki/Social_network

[81] http://jade.tilab.com/ The Java Agent Development Framework

[82] http://en.wikipedia.org/wiki/Q_learning

[83] Yang B., Garcia-Molina H., “Designing a Super-Peer Network”. Proceedings of the 19th International Conference on Data Engineering (ICDE), Bangalore, India, March 2003.

[84] Xiong L. and Liu L., “PeerTrust: Supporting Reputation-Based Trust for P2P Electronic Communities”. IEEE Transactions on Knowledge and Data Engineering, Vol.16, No. 7, Special issue on Peer to Peer Based Data Management, July 2004, pp. 843-857.

[85] Yolum P. and Singh P. M., “Emergent Properties of Referral Systems”. Proceedings of the 2nd International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS), July 2003, pp. 592–599.

[86] Yu B. and Singh P. M., “A Social Mechanism of Reputation Management in Electronic Communities”. Proceedings of the 4th International Workshop on Cooperative Information Agents IV, The Future of Information Agents in Cyberspace, 2000, pp. 154-165.

[87] Yu B., Singh P. M,, and Sycara K., “Developing Trust in Large-Scale P2P Systems”. Proceedings of First IEEE Symposium on Multi-Agent Security and Survivability, 2004, pp. 1-10.

[88] Zacharia G. and Maes, P., “Trust Management through Reputation Mechanisms”. Applied Artificial Intelligence, vol. 14, 2000, pp. 881-907.

[89] Zhang J. and Cohen R., "Design of a Mechanism for Promoting Honesty in E-Marketplaces". Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07); July 2007, pp. 1495-1500.

[90] Zhang J. and Cohen R., " Evaluating the Trustworthiness of Advice about Selling Agents in E-Marketplaces: A Personalized Approach”. Journal of Electronic Commerce and Research Applications, Volume 7, Issue 3, 2008, pp. 330-340.

[91] Zhang Y. and Larson K., “Reducing Interaction Cost: A Mechanism Design Approach”. Lecture Notes in Business Information Processing 14, Springer, 2009, pp. 166–181.

[92] Zhang Y., Lin J. K., and Klefstad R., “DIRECT: A Robust Distributed Broker Framework for Trust and Reputation Management”. Proceedings of the IEEE International Conference on E-Commerce Technology and Enterprise Computing, E-Commerce, and E-Services, 2006.